Contents lists available at ScienceDirect

Computational and Structural Biotechnology Journal

journal homepage: www.elsevier.com/locate/csbj

Research Article

Predicting protein complexes in protein interaction networks using Mapper and graph convolution networks

Leonardo Daou, Eileen Marie Hanna *

Department of Computer Science and Mathematics, Lebanese American University, Byblos, Lebanon

ARTICLE INFO

Keywords:
Protein-protein interactions
Protein complex
Topological data analysis
Generative adversarial network
Graph convolutional network
Mapper algorithm

ABSTRACT

Protein complexes are groups of interacting proteins that are central to multiple biological processes. Studying protein complexes can enhance our understanding of cellular functions and malfunctions and thus support the development of effective disease treatments. High-throughput experimental techniques allow the generation of large-scale protein-protein interaction datasets. Accordingly, various computational approaches to predict protein complexes from protein-protein interactions were presented in the literature. They are typically based on networks in which nodes and edges represent proteins and their interactions, respectively. State-of-the-art approaches mainly rely on clustering static networks to identify complexes. However, since protein interactions are highly dynamic in nature, recent approaches seek to model such dynamics by typically integrating gene expression data and identifying protein complexes accordingly. We propose MComplex, a method that utilizes time-series gene expression with interaction data to generate a temporal network which is passed to a generative adversarial network whose generator is a graph convolutional network. This creates embeddings which are then analyzed using a modified graph-based version of the Mapper algorithm to predict corresponding protein complexes. We test our approach on multiple benchmark datasets and compare identified complexes against gold-standard protein complex datasets. Our results show that MComplex outperforms existing methods in several evaluation aspects, namely recall and maximum matching ratio as well as a composite score covering aggregated evaluation measures. The code and data are available for free download from https://github.com/LeonardoDaou/MComplex.

1. Introduction

Proteins are vital to various cellular functions in all living organisms. They consist of a polypeptide chain of amino acids that code for the genetic information of genes. The tertiary structure of a protein defines its three-dimensional shape and subsequently the precise placement and interaction with specific chemical groups to perform certain functions. For instance, proteins act as enzymes and hormones to break down ingested food and regularize growth and reproduction. They are also involved in healing, regeneration, and production of antibodies [1]. Some proteins can function alone. However, proteins often form large macromolecular complexes by interacting with each other to coordinate and carry out various molecular functions within the cell [2,3]. Identifying such entities is not only essential in understanding the principles of cellular organization and function mechanisms, but also in elucidating malfunctions that lead to diseases [2,4,5]. For example, the prediction of protein complexes aided to better understand biological processes related to including human immunodeficiency viruses (HIV) [6] and cancer [7], as well as general processes like apoptosis [8].

Large-scale protein-protein interaction (PPI) data can be obtained from high-throughput experimental techniques such as mass spectrometry and yeast two-hybrid [9,10]. Subsequently, multiple computational approaches were designed to identify protein complexes from such datasets. They are typically based on a PPI network in which the proteins and their interconnections are represented by nodes and edges, respectively. State-of-the-art methods predominantly rely on clustering static interaction networks. We hereby highlight several such methods. ClusterONE [11] chooses nodes that have the highest degree in the network for cluster generation. These nodes are called seeds and are grown into clusters by considering the surrounding of each node of each cluster. This results in overlapping clusters as included nodes can be repeated in different expansions. Similar to ClusterONE, NCMine

* Corresponding author.
E-mail address: eileenmarie.hanna@lau.edu.lb (E.M. Hanna).

https://doi.org/10.1016/j.csbj.2024.10.009
Received 10 June 2024; Received in revised form 4 October 2024; Accepted 4 October 2024
Available online 10 October 2024

[12] begins by selecting seeds from which clusters are generated. Clusters are expanded if the cliqueness value-a similarity measure between a cluster and a complete graph-exceeds a certain pre-defined threshold after adding a node. Cliqueness is the similarity between a cluster and a complete graph. The core-attachment based method (COACH) [13] constructs a neighborhood graph G by removing all nodes with a degree of 1 from the input network and selects core vertices, which have a degree at least equal to the average degree of G. The neighborhoods of these core vertices are taken as preliminary cores, which are refined into protein complex cores by removing the core vertices and adding them back to the connected components that result from their removal. The protein complex cores are then expanded by merging neighboring vertices if they are connected to more than half of the core's internal nodes. The Molecular Complex Detection (MCODE) approach [14] obtains clusters based on node weights, using the notion of k-core to determine these weights. A k-core is a subgraph where each node has a degree equal to or bigger than k. As such, the weight of the node is the highest k-core that includes it. Nodes with the highest weight are assigned as seeds. The seeds are then expanded by including nodes that have a weight above a certain threshold. DPClus [15] uses an alternative definition for node weight which is the sum of the weights of the edges that are connected to it. The weight of an edge is defined as the number of common neighbors that its corresponding nodes have. Seeds are chosen and clusters are grown by considering neighboring nodes according to priority which is based on the sum of the weights between the neighbor and the nodes in the community as well as the number of edges between them.

In contrast, the clustering-based on maximal cliques (CMC) method [16] computes the protein-protein interaction weight, representing the density of the region in the PPI network where the interaction occurs. It then identifies all maximal cliques—those that cannot be expanded by adding adjacent vertices—and ranks them by decreasing density. Overlapping cliques are either merged or removed while keeping the highest-scoring clique. This decision is based on the connectivity between the non-overlapping parts of the cliques. The clustering based on network structure and ontology attribute similarity (CSO) approach [17] also incorporates maximal cliques but adds gene ontology (GO) information to the PPI network. Since each protein can participate in multiple biological processes, this method identifies maximal ontology-correlated cliques, which are cliques formed by proteins sharing the same GO information. From these, seed cliques—used to form predicted protein complexes—are chosen based on an ontology correlation score, reflecting the relationship between the number of proteins in the clique and their shared GO data. The seed cliques are expanded by adding neighboring nodes that meet a certain GO similarity threshold. CFinder [18] relies on the fact that protein complexes mostly correspond to dense regions and are even more likely to be cliques. Hence, it uses a concept called k-clique percolation cluster which is formed by the nodes reachable from each other through a chaining of adjacent k-cliques as well as the edges in those cliques. In other words, a community of adjacent k-cliques forms a cluster. Contrary to the previous usual clustering algorithms, Neural Overlapping Community Detection-Graph Convolutional Network (NOCD-GCN) [19] uses a Graph Convolutional Network (GCN), typically a supervised learning approach, in an unsupervised manner to predict protein memberships in various complexes. Instead of relying on known complex labels for training, the method utilizes the membership probabilities estimated by the Bernoulli–Poisson (BP) model, which is based on the adjacency matrix. The model aims to adjust these predicted membership probabilities to ensure they are consistent with the observed edges in the graph, thereby aligning the inferred community structure with the actual protein interactions.

The described approaches only utilize the PPI network as the initial data. However, the properties of proteins can change depending on certain characteristics and their potential interactions can thus vary [20]. Multiple approaches were proposed for protein complex prediction using dynamic PPI networks that are obtained from time-series gene expression data. A time-series dataset reflects observation values across multiple time-points. Gene expression data is typically chosen for integration with the PPI networks as it describes the different expression levels of each gene at different times. We discuss multiple such methods.

Zhang et al. [21] proposed an integration approach that calculates active probabilities which reflect whether certain proteins and interactions are present at a certain time-step. Given that each gene corresponds to a certain protein, then each time-step in the gene expression dataset has a different network. These newly obtained networks are grouped together to form a dynamic network with each subnetwork corresponding to a different time-point. Similar to previous methods, they utilized the idea of a seed from which a cluster is grown. However, instead of taking a seed node, a seed edge is considered. Seed edges are chosen based on a score representing the number of common neighbors shared by the nodes forming the edge. For each seed, the neighbors of the nodes that are linked by the edge are added to form a core, if their addition to the seed results in a cluster is of density higher than a certain threshold. Afterwards, for each core, its neighboring proteins are added if their attach score, which is a measure indicating the membership of the protein in the considered core, is above its current density. This results in a set of clusters for each subnetwork of the dynamic network. They are joined to form a set of predicted protein complexes. DDH [22] also integrates gene expression with PPI data to obtain a temporal PPI network through the active probability of a protein and its interactions at specific time intervals. For each subnetwork, the approach forms initial clusters based on a cluster center, which is similar to a seed. Cluster centers are nodes chosen according to the density formed by the subgraph including it and its neighbors as well as the distance between two high-density nodes. The functional annotation of a protein is used to obtain such distance. Furthermore, cluster centers are chosen to maximize the distance between them to avoid overlapping dense regions. Once the centers are chosen, non-central nodes are added to the nearest high-density cluster to obtain the initial clusters. Then, a heuristic algorithm is used to move proteins from one cluster to another. CPredictor3.0 [23] produces the dynamic network in a similar manner as DDH. Once the network is obtained, the approach computes the functional similarity between two proteins for each subnetwork. Since a protein can be a part of multiple biological processes, the functional similarity between a pair of proteins is delineated as the number of biological processes in which both are present together. Given the original PPI network and through spectral clustering, the similarity matrix is clustered to form communities of similarly functional proteins. The intersection between each subnetwork's active proteins and the functional protein clusters is taken to obtain active protein clusters of similar function for each time-step. Each active protein cluster of similar function is projected to the subnetwork, and every connected component containing more than two nodes is considered a candidate complex.

In the context of predicting protein complexes, it is challenging to identify such complexes while ensuring they align with a high number of known complexes and also maintaining similar structures between the two. Hence to increase these two aspects, we present MComplex, an approach that identifies protein complexes from a temporal network which integrates time-series gene expression with protein-protein interaction data. It obtains a representation of the network through a graph convolutional network (GCN) trained by a generative adversarial network (GAN) [24]. This embedding is passed through a modified graph-based version of the Mapper algorithm [25,26] which outputs the predicted protein complexes. In the next sections, we start by explaining the integration performed on the two types of data, and we present a comprehensive description of the proposed methodology. Then, we report the experimental results in which MComplex is evaluated against several state-of-the-art approaches. Finally, we discuss the results and highlight potential future enhancements in the Discussion and Conclusion sections, respectively.

Table 1
Statistics of the PPI datasets used with the values representing: number of proteins, number of interactions.

PPI Dataset	Static Network	Dynamic Network Time-Steps					
		T1	T2	T3	T4	T5	T6
Gavin Dataset	(1430,6531)	(780,2072)	(529,1137)	(381,733)	(960,4104)	(1019,4399)	(626,1426)
Krogan Dataset	(2675,7080)	(1529,3028)	(1047,1784)	(739,1060)	(1563,3519)	(1569,3763)	(1167,2004)
MIPS Dataset	(3950,11119)	(2328,5298)	(1761,3681)	(1269,2109)	(2290,5144)	(2172,5217)	(1801,4169)
STRING Dataset	(5714,104574)	(3617,35517)	(2755,22529)	(2191,12558)	(3615,60474)	(3368,60298)	(2918,23886)

2. Methods

2.1. Temporal network construction

Various PPI datasets are generated using high-throughput experimental techniques. They typically consist of a set of protein pairs that interact with each other. Accordingly, we can generate a static PPI network in which nodes and edges represent proteins and their interactions, respectively. In order to model the dynamic nature of protein interactions, we integrate time-series gene expression data which represents the expression levels of genes at different time-steps. By doing so, we obtain a temporal PPI network following the work by Wang et al. [20] and Zhang et al. [21]. We use four benchmark yeast PPI datasets: Gavin [27], Krogan [28], MIPS [29] and STRING [30]. We note that STRING is one of the largest PPI datasets since it covers yeast PPI data obtained from biomedical literature data, co-expression data, high-throughput data, and genomic context data. Accordingly, in this dataset, we only consider interactions that have a confidence score that is above 0.8. The number of proteins and their interactions in each dataset are shown in Table 1.

We also select GSE3431 [31] from the Gene Expression Omnibus (GEO) [32] as the time-series gene expression dataset. This dataset is obtained by array affymetrix and it includes 9335 probes. Its experimental design is divided into 3 cycles, with each cycle consisting of 12 time intervals. Each time interval records 25 minutes. In other words, each cycle contains 12 time-points for each gene. We average the expression levels at time-points T_{ij}, T_{ij+1} and T_{ij+2}, where i is the time-step in cycle j, to derive a summarized expression level for the ith time-point. Through this, the initial 36 time-steps are reduced to 12. For gene expression data, higher expression levels indicate the increased production and presence of a certain protein. According to [20], instead of relying on a static PPI network, and in order to model such dynamics, it is crucial to figure out not only the time-points where the proteins are present, but also the time-points where these proteins are active. This is due to the fact that the presence of a protein does not necessarily mean that it is active and part of a complex. Furthermore, for two protein to form an interaction, they both need to be active at the same time point in addition to them being compatible which is known from the static PPI network. Thus, Wang et al. [20] assumed that proteins are active at the time-points with the highest expression levels, especially since the expression level of an active protein decreases after its function is completed. In this context, we assume a normal distribution for the GSE3431 gene expression datasets and adopt multiple expression thresholds, specifically the three-sigma method, as done in [20] and [21]. This is needed to differentiate between active and inactive proteins at different time-steps. Wang et al. [20] proved its effectiveness by correctly locating almost every active essential protein given by the integration between a static PPI network and gene expression data. We use the same approach for identifying active time points for each proteins and follow the subsequent step of the integration method proposed by [21]. The k-sigma threshold [33] can be calculated through

$$Thresh_k(p) = \alpha(p) + k.\sigma(p).\left(1 - \frac{1}{1+\sigma^2(p)}\right) \tag{1}$$

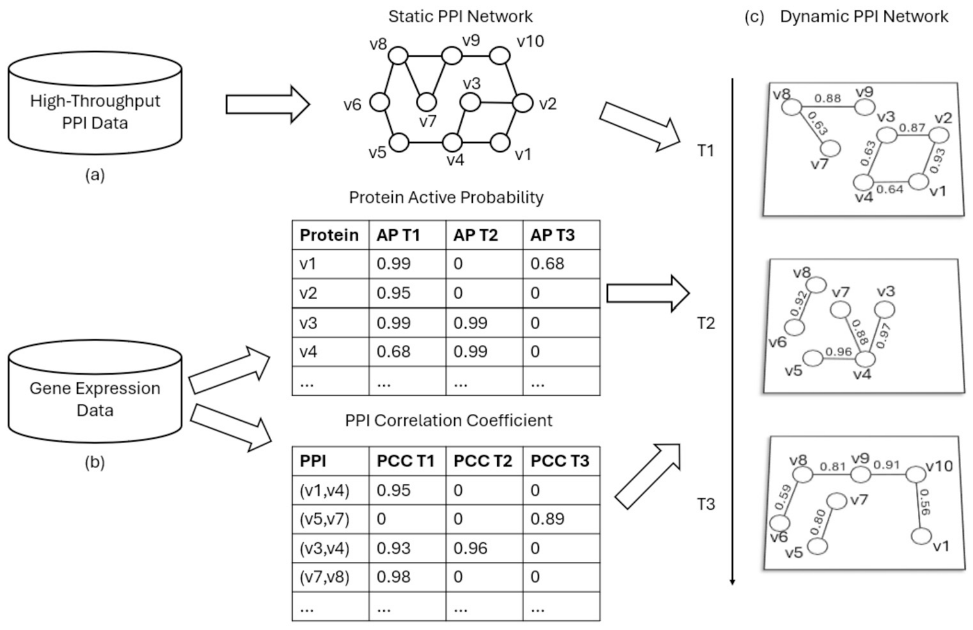
where p represents a gene expression observation, $\alpha(p)$ represents the arithmetic mean of p, $\sigma(p)$ represents the standard deviation (SD) of p. For the three-sigma method, k takes the values {1,2,3}. Given a real random variable X of normal distribution with mean α and variance σ^2, the probability that X exists in a certain setting given a certain k would be $P\{|X-\alpha| < k\sigma\} = 2\Phi(k) - 1$ for which Φ represents the distribution function of the standard normal law. Furthermore, based on the chosen k values, this also corresponds to the empirical rule of the normal distribution where, for $k=1$, $P\{|X-\alpha| < \sigma\} = P\{\alpha-\sigma < X < \alpha+\sigma\} = 0.6827$. The same is applied for $k=2$ with a probability of 0.9545 and for $k=3$ with a probability of 0.9973. Hence, as the value of k increases, the likelihood that X falls within the range of the specified distribution also increases. Compared to the $Thresh_k$ criterion given by Equation (1), the higher the k value, the higher the threshold value that marks that a protein as active. Given that $G_i(p)$ represents the gene expression value of a certain gene p at time-step i, then when $G_i(p) > \alpha(p) + 3.\sigma(p)$, the probability that p is active at the ith time-step is 99.7%. However, when k is 1, then the threshold is lower, but so is the confidence that the protein is actually active at a certain time-point. That is the case since the probability is not 68.3%. Through the empirical rule expanded on above, the active probability $PR_i(p)$ [21] is given by

$$PR_i(p) = \begin{cases} 0.99 & \text{if } G_i(p) \geq Thresh_3(p) \\ 0.95 & \text{if } Thresh_3(p) > G_i(p) \geq Thresh_2(p) \\ 0.68 & \text{if } Thresh_2(p) > G_i(p) \geq Thresh_1(p) \\ 0 & \text{if } G_i(p) < Thresh_1(p) \end{cases} \tag{2}$$

where the first three levels depict the probability of a protein p being active at time-step i, given that its corresponding expression level $G_i(p)$ clears the kth threshold. Hence, if $G_i(p)$ is lower than all three thresholds, p is marked as inactive for time-step i, and the probability is then 0. This approach improves on previous ones due to the fact that it identifies active time-points for each protein as well as a corresponding probability that p is active at time-step i. Given that this should be done for proteins who are depicted in a network, then the activity PPI network Act_i at time-step i is given by

$$Act_i = Pr_i Pr_i^T \tag{3}$$

where the activity of all proteins at time-point i is represented by the column vector Pr_i, and Pr_i^T is its transpose. Furthermore, given that there might some differences in the expression levels across the three successive cycles in GSE3431, we averaged the expression values across these cycles and we also applied the three-sigma rule which help mitigate this issue [20]. Afterwards, the interactions given by the static PPI network are integrated through the calculation of gene co-expression. We use the co-expression correlation coefficient which measures the similarity of expression variance patterns between co-expressed genes across different conditions. Such measure indicates a strong signal when two proteins have functional associations, and hence, most likely belong in the same complex. Additionally, the co-expression value is also integrated with the protein active probabilities and the interaction data from the static PPI network. We note that if two proteins have a high co-expression value and interact in the static PPI network, they can be inactive at a certain time-point after completing certain cellular functions [20]. Hence, the three sources of information complement each other and can help remove any potentially non-existent protein-protein interaction at a certain timestep.



Protein Active Probability

Protein	AP T1	AP T2	AP T3
v1	0.99	0	0.68
v2	0.95	0	0
v3	0.99	0.99	0
v4	0.68	0.99	0
...	...	...	...

PPI Correlation Coefficient

PPI	PCC T1	PCC T2	PCC T3
(v1,v4)	0.95	0	0
(v5,v7)	0	0	0.89
(v3,v4)	0.93	0.96	0
(v7,v8)	0.98	0	0
...	...	...	...

Fig. 1. Integration of PPI networks and gene expression data. (a) Static PPI networks' construction from high-throughput PPI data. (b) Computation of dynamic information using gene expression data with T representing a certain time-point, AP representing active probability and PCC representing Pearson correlation coefficient. (c) Dynamic PPI networks' construction. Adapted from Zhang et al. [21].

In order to obtain the correlation matrix between the gene expression levels, the normalized Pearson correlation coefficient is used [34]. Given that gene co-expression data also reflects protein co-expression data, then the same operation is performed to obtain the co-expression protein networks *Coe* with Coe_i being the co-expression protein network for time-step *i*.

The calculation of a correlation matrix here considers multiple time-points. Thus, a sliding window that covers three consecutive time-points {*i-1*, *i* and *i+1*} is chosen for time-step *i*. To remove correlation coefficients that have a small value in Coe_i, a predefined threshold Pre_{th} is used:

$$Coe_i(m,n) = \begin{cases} P_{Cor_i}(m,n) & \text{if } P_{Cor_i}(m,n) \geq Pre_{th} \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where the Pearson correlation coefficient (PCC) between protein *m* and protein *n* for time-step *i* is given by $P_{Cor_i}(m,n)$. Based on [21], we set the predefined threshold Pre_{th} to 0.5. Subsequently, through the integration of the activity in PPI networks, the co-expression protein matrix, and the static PPI network *adjSPN*, the dynamic PPI networks $adjDPN_i$ can be constructed through:

$$adjDPN_i = Act_i \circ Coe_i \circ adjSPN \tag{5}$$

with the three different matrices undergoing element-wise multiplication to obtain the weight of the interaction between two proteins at time-step *i*. Once applied on GSE3431, the sliding window of step size 2 is used to obtain the corresponding correlation matrix. This means that once the inputs at $\{i-1, i, i+1\}$ are aggregated, they result in one time-step *k* represented by a subnetwork $adjDPN_k$ of the temporal network adjDPN obtained from the integration of the two types of datasets as mentioned in Section 2.1. Next, the window then slides to the values at $\{i+1, i+2, i+3\}$. This results in a dynamic PPI network with 6 time-steps. Such integration is done for the Gavin, Krogan, MIPS, and STRING datasets to obtain Gavin_DPN, Krogan_DPN, MIPS_DPN, STRING_DPN. The number of proteins and their interactions for each time-step are shown in Table 1. It can be seen that the active proteins change based on the time-step with some containing more proteins and interactions. The increase and decrease in the number of proteins perceived between time-steps aligns with the observations made by Wang et al. [20] where proteins expression levels reduce over time until they become inactive.

This construction process is shown in Fig. 1. Given proteins v_3 and v_4, the static PPI network represents the presence of an interaction by 1 and 0 otherwise. To obtain the dynamic PPI network's edge for these two proteins at time-step *i*, both should be active according to the protein active probability. Such interaction must have a correlation coefficient higher than Pre_{th} for that specific time-step. Given that there is an interaction between v_3 and v_4, they both are only active on time-steps T_1 and T_2 based on the active probabilities. Both time-steps also have a high correlation coefficient between them. Hence, applying Equation (5) gives their interaction probabilities for both time-steps with 0.63 for T_1 and 0.97 for T_2.

2.2. *Protein complex prediction*

2.2.1. *Cluster formation*

The Mapper algorithm [35] is a useful method for high-dimensional data visualization and for the calculations of data embeddings that can be used as input for other methods. One such application is clustering. Accordingly, our proposed approach, MComplex, utilizes the Mapper algorithm to computationally obtain potential protein complexes from a dynamic PPI network. However, given that its typical application only takes a data point cloud with no interconnections, we utilize a modified graph-based version of this algorithm [25].

The Mapper algorithm only takes as input one static network at a time, it is thus applied on each time-step separately. Each subnetwork undergoes the following steps.

Lens function First, a lens function, i.e. a filtering function $f: V \rightarrow \mathbb{R}^d$, is defined. It is typically a filter that focuses on specific aspects of the graph. As can be seen in Fig. 5(a), it creates a range of values that represent the vertices. Given that protein complexes consist of interacting proteins, a fitting lens function can be seen as an embedding of the network that represents these interconnections as well as dense regions of the graph. Hence, we use a Graph Convolutional Network (GCN) which can naturally give patch representations of a node and its neighborhood [24]. A GCN takes as input a network of interest as well as the features of each node. Since we are interested in the protein interactions, we use

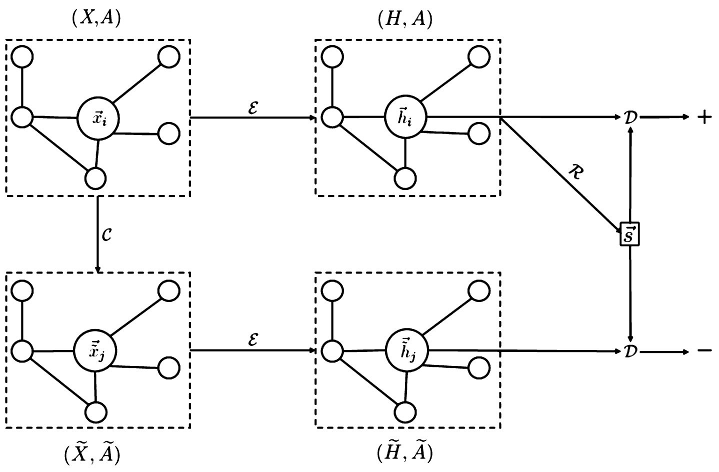


Fig. 2. Overview of Deep Graph Infomax. Adapted from Veličković et al. [24].

the identity matrix I. It is an nxn matrix in which the principal diagonal's elements are ones and the rest of the elements are zeros. However, since GCN is a supervised learning method, a Generative Adversarial Network (GAN) is used to train it. Specifically, the Deep Graph Infomax (DGI) model [24] is utilized. Given that the objective is to obtain a graph representation, DGI is made of a generator, also called an encoder, which is the GCN as well as a discriminator D which can assess the obtained representation and backpropagate the error. GANs require positive and negative examples to be able to train the generator. Thus, taking the inputted network's vertices as positive examples, negative examples are generated by modifying the network through a corruption function $C(X,A)$ where X is a matrix where each row corresponds to a node's features and A is the adjacency matrix of the network. Corruption refers to shuffling an aspect of the network. In this case, X undergoes row-wise shuffling which results in each node randomly being assigned a different row of features. No row is repeated in the shuffling process. Subsequently, the corrupted network nodes are labeled as the negative samples. This process also creates a balanced dataset for the GAN where 50% of the data consists of positive examples and the other 50% represents the negative examples.

Following Fig. 2, after the inputted graph is corrupted, both positive and negative samples are passed through the encoder to obtain the embedding, i.e., patch representations. Those representations for the positive examples are written as $H = \mathcal{E}(X,A) = \{\vec{h}_1, \vec{h}_2, ..., \vec{h}_N\}$ with $\vec{h}_i$ being the features of node i that represents its surrounding neighborhood and N being the number of nodes in the original graph. The embeddings for the negative examples are written as $\widetilde{H} = \mathcal{E}(\widetilde{X}, \widetilde{A}) = \{\vec{\tilde{h}}_1, \vec{\tilde{h}}_2, ..., \vec{\tilde{h}}_M\}$ with $\vec{\tilde{h}}_j$ being the features of node j that represent its surrounding neighborhood and M being the number of nodes in the corrupted graph. Afterwards, to be able to train the encoder to give the best representation of the original graph, a summary vector $\vec{s}$ of the node embeddings of the original graph is needed. This vector is obtained through a readout function which can be written as:

$$\mathcal{R}(H) = \sigma\left(\frac{1}{N}\sum_{i=1}^{N}\vec{h}_i\right) \tag{6}$$

where σ is the sigmoid activation function. This readout function takes as input the patch representations of a graph and averages all of the nodes features and then squashes the values between 0 and 1. Subsequently, the summary vector $\vec{s}$ is given to the discriminator along with the node embedding for evaluation. The discriminator, which is a simple bilinear scoring function, is given by:

$$\mathcal{D}(\vec{h}_i, \vec{s}) = \sigma(\vec{h}_i^T W \vec{s}) \tag{7}$$

where W is a learnable weight matrix, and σ is the sigmoid activation function. Through the summary vector of the original graph given, the output of the discriminator is the likelihood that a certain node is part of the positive example. This means that the objective of the GAN is to maximize the output of the discriminator to 1 when the node embedding passed is from the original graph, and minimize it to 0 when the patch representation passed is from the corrupted graph. To optimize the weight matrix of the discriminator as well as the weights, the loss function used is given by:

$$\mathcal{L} = \frac{1}{N+M}\left(\sum_{i=1}^{N} E_{(X,A)}\left[log\mathcal{D}(\vec{h}_i, \vec{s})\right] + \sum_{j=1}^{M} E_{(\widetilde{X},\widetilde{A})}\left[log(1-\mathcal{D}(\vec{\tilde{h}}_j, \vec{s}))\right]\right) \tag{8}$$

where $E_{(X,A)}\left[log\mathcal{D}(\vec{h}_i, \vec{s})\right]$ represents the expected values of the loss function for the ith probability distribution of the network node features, while $E_{(X,A)}\left[log\mathcal{D}(\vec{h}_i, \vec{s})\right]$ is the expected values of the loss function for jth corrupted network's node features probability distribution. Expected values of these functions are used based on the fact that generators attempt to approximate probability distributions and not only the given values. This loss is then backpropagated through the discriminator and the generator.

The GCN employed for the encoder is built based on three GCN layers with the first one having 32 neurons, the second one having 64 neurons, and the last one having 512 neurons. Every GCN layer added to the neural network allows a node's features to encompass a bigger patch of the inputted graph as its representation as shown in Fig. 3. The three graph representations have the same connectivity, but each node's representation is different, as seen in the previous explanation. Thus, they can be designated as $G = (V,E)$. Such representations follow the range of the patch summary around the node colored yellow X. For the first hidden layer represented in Fig. 3(a), the patch summary takes into account the immediate neighborhood, represented by the first circle, to obtain a new vector representation for node X. The immediate neighborhood of node X can be written as $\{v \in V\,|v \in Adj(X)\}$ where $Adj(X)$ is the adjacent nodes of X. The same convolutions are done for all of the nodes in the graph. Afterwards, another circle is added which depicts what node X now represents in Fig. 3(b). It is shown that it now represents its immediate neighbors as well as its neighbors' neighbors, which can be written as $\{u \in V\,|u \in Adj(v)\}$. This is due to the fact that the same operation is applied on all of the nodes at the same time. The latter means that the neighbors' representation takes into account their own adjacent nodes. Hence, when the aggregation operation is done for node X in the second hidden layer, even though the operation only takes into account nodes v, it is also affected by nodes v's neighbors u. As more hidden layers get added to the neural network, the summarized patch gets bigger. This is the case for the network in Fig. 3(c) where another circle to rep-

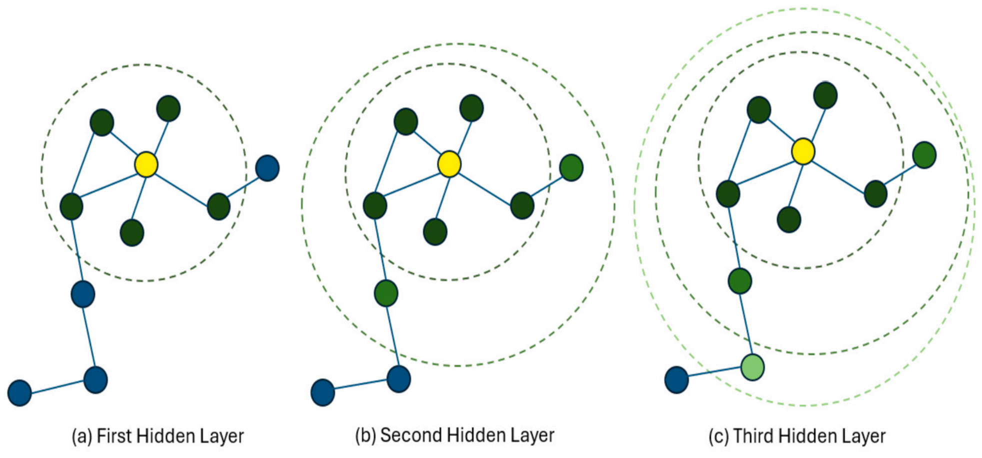


Fig. 3. Graph Convolutions at (a) the first hidden layer, (b) the second hidden layer and (c) the third hidden layer.

resents the new neighborhood in the patch. As was the case in Fig. 3, the neighbors of the peripheral nodes are added in the second hidden layer which were nodes u.

In our approach, the outputs of the first two GCN layers are passed through an exponential linear unit (ELU) [36] activation function which is given by:

$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^x - 1) & \text{otherwise} \end{cases} \quad (9)$$

where α is a hyperparameter that is usually set to 1.0, which is also the case for the current encoder. The GAN is then trained for 1000 epochs with the number of epochs being set experimentally. Loss curves over 1000 epochs for the four dataset obtained from masking 80% of the nodes for the train loss and the other 20% for the test loss are shown in supplementary figures 1 to 4. We can observe a significant improvement across the 1000 epochs for the Gavin and Krogan dynamic networks. The improvement is still noticeable on the STRING network as some of the timesteps obtain the same improvement rate as the former two datasets while others stagnate shortly after training commences. This stagnation is also noticed when the GAN is applied on the MIPS dynamic network along with some fluctuation to the loss suggesting that the gradient gets stuck in a local minimum shortly after training starts. However, the overall performance is satisfactory as the application of the proposed GAN on most of the timesteps across the four dynamic networks achieve substantial results in terms of loss. The embeddings generated also contribute in the results shown in Section 3.

After the training of the GAN is done, the original graph is then passed to the encoder to obtain an embedding with 512 features for each node. Since the Mapper algorithm usually takes a low dimensional lens to be computationally efficient, the dimension of the obtained embedding is reduced using Uniform Manifold Approximation and Projection (UMAP) [37] to get a two dimensional representation of the graph that is used as the lens. UMAP preserves the topological structure of the data and the distance between the data points which is suitable since its output dictates the initial clustering of the proteins.

Lens cover Once the representation is obtained, the nodes are grouped into n equal-size intervals that correspond to subgraphs of the original network as visualized in Fig. 5(b). That is based on the range of values obtained from the filter functions. The elements in each interval are called cover elements I. The bigger the value of n, the more clusters are obtained since more groups are created. Added to that, an overlap percentage needs to be set. It specifies how much these intervals can intersect as shown in Fig. 5(c). These two values are hyperparameters that need to be tuned. Hence, a grid search is performed to obtain their most optimal values on each dataset. For the number of intervals, a sequence of values between 10 and 80 with a step of 5 is chosen, while

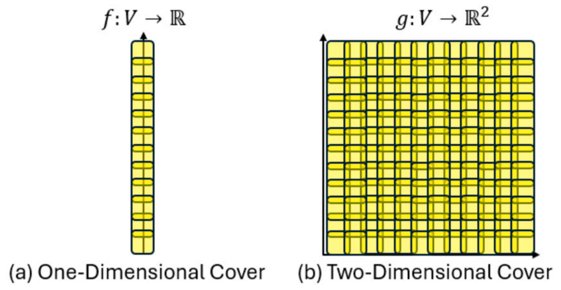


Fig. 4. Covers for two different lenses. (a) Cover for a one-dimensional lens. (b) Cover for a two-dimensional lens.

another sequence ranging from 20 to 90 with a step of 10 is chosen for the overlap percentage. For each dataset, we obtain a different set of optimal parameter values for the Mapper algorithm. The range of values for the number of intervals and the overlap percentage is chosen to be large enough while also taking into consideration the computational time needed. This arises from the fact that as the number of intervals increase, the Mapper Algorithm needs to examine and cluster an increasingly large number of subgraphs of the network. Furthermore, given that the lens function consists of two dimensions, unlike what is represented in Figs. 4(a) and 5(a), the cover splits the network across a two-dimensional plane with $n \times n$ intervals as shown in Fig. 4(b).

Inverse image An inverse image [25] needs to be computed for each cover element. It is the subset of nodes of V in which lens function values correspond to the cover element. These constitute a subgraph of the original network which is then clustered in the next step. Some nodes are duplicated in other cover elements due to the nature of the Mapper algorithm in which overlaps are allowed. In order to form the mapper nodes from the cover elements as depicted in Fig. 5(d), an appropriate clustering algorithm is needed.

Graph clustering algorithm We use Label propagation [38] as the clustering algorithm for each cover element in our proposed approach. It assigns each node to the major community in its neighborhood. The algorithm begins by assigning a label to each node x, where each label represents one cluster. Then, it randomly chooses one vertex at a time and assigns the label that occurs at the highest frequency among that node's neighbors. The stopping condition is met at the end of the iteration if each node possesses the label that is the most recurring among its neighbors, at which point the communities are final. The resulting clusters from all cover elements are the preliminary protein complexes.

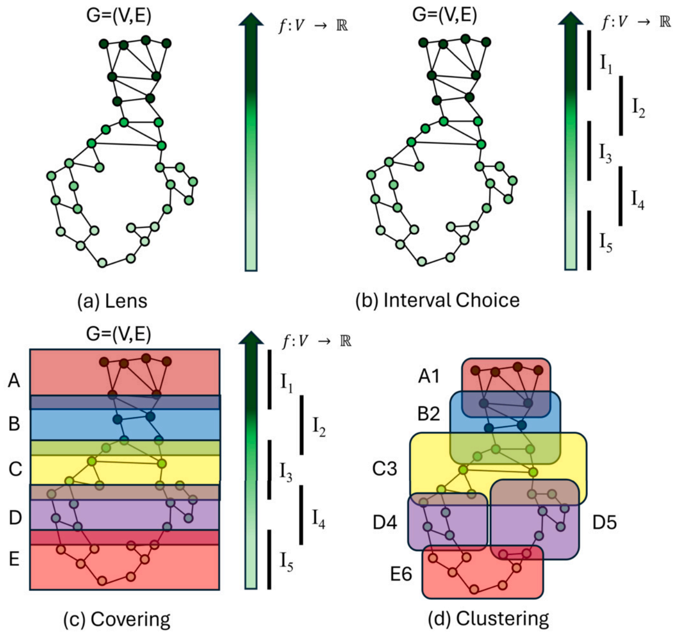


Fig. 5. Workflow of Mapper on Graphs. (a) Choosing the lens function that represents each node. The lens function *f*, which is one dimensional, is represented by the gradient-filled arrow. (b) Choosing the number of intervals as well as the overlap percentage between them. (c) Applying the cover on the graph. (d) Applying a clustering algorithm on each of the intervals. Adapted from Loughrey et al. [35].

2.3. Post-processing

Given that each $adjDPN_i$ undergoes clustering on its own, the resulting clusters need to be joined together to form the final complexes. The method adopted to group these clusters is inspired by ClusterONE [11]. The clusters obtained from the different subnetworks are joined into one set S_1, and the remaining proteins that are present in the original PPI network but not in any of the time-steps are added to another set S_2. All of the singular clusters, i.e. clusters of size 1, are removed from S_1, and the proteins that comprise them are added into set S_2. Duplicate proteins in set S_2 are allowed in order to keep the overlapping property. The clusters remaining in set S_1 are then considered one at a time to check if the removal of any protein in it would increase its cohesiveness. If such protein is found, it is removed from the cluster being studied and added to set S_2. Cohesiveness [39] is proportional to the density of a cluster and is given by:

$$Cohesion(C_i) = \frac{w_{in}(C_i)}{w_{in}(C_i) + w_{out}(C_i) + p} \tag{10}$$

where C_i is the *i*th cluster, w_{in} is the sum of the weights of interconnections for the *i*th cluster, w_{out} is the sum of the weights of interconnections between the *i*th cluster and the proteins in the rest of the network, and *p* is a penalty term indicating the uncertainties in the PPI network. After all clusters are checked, they are then sorted by their decreasing order of outgoing edges. The proteins in set S_2 are also sorted by their decreasing order of degree. The method then checks every cluster and tries to add to it the proteins from S_2 one at a time. If the protein increases the cohesiveness, then it is kept. Once all clusters are considered, the remaining proteins in S_2 are discarded. Following that, the clusters from set S_1 are then considered as nodes in a network. In this case, two nodes are connected by an edge if there exists an overlap between them higher than a certain threshold th_{over}. The overlap between two clusters is obtained [11] by:

$$Overlap(A,B) = \frac{(A \cap B)^2}{|A| \times |B|} \tag{11}$$

where *A* and *B* are two clusters. An overlap between two clusters occurs when a certain amount of elements in one cluster is present in the other one. Once the network is formed, the clusters in the connected components are merged while only keeping one version of each protein in the case of duplicates. A threshold of 0.5 is chosen based on the empirical results obtained after testing multiple threshold values ranging from 0.4 to 0.9 with a step of 0.05. In other words, the clusters that have an overlap of more than 0.5 are merged together. We chose 0.4 as the lower bound of the range since lower values would merge clusters that are dissimilar enough to count as different predicted clusters, and the interconnections would not be related. They also cause abnormally large clusters to be found in the predicted complexes outputted. This threshold has also been used in previous works such as in CMC [16]. No significant protein interactions are lost in this process.

Furthermore, among the resulting clusters, some might be extremely sparse. In other words, they contain very few interconnections which contradicts the main assumption that protein complexes mostly correspond to densely-connected clusters. Hence, a density threshold th_{dens} is imposed on the identified clusters to discard such complexes. Any predicted cluster that has a density less than th_{dens} is removed. The density of a cluster [21] is given by:

$$Density(C_i) = \frac{2 \times \sum_{e(u,v)\in E_{C_i}} adjSPN(u,v)}{|V_{C_i}| \times (|V_{C_i}| - 1)} \tag{12}$$

where C_i is the *i*th cluster, *u* and *v* are two nodes of *adjSPN*, E_{C_i} is the set of internal edges of C_i, and V_{C_i} is the set of nodes of C_i. The threshold of 0.2 was empirically chosen from a range of values between 0 and 1 with a step of 0.1. Once the clusters that do not meet the threshold are removed, then the remaining clusters in set S_1 form the output protein complexes obtained from the proposed method.

2.4. Evaluation metrics

In order to measure the performance of our proposed method, the predicted complexes are matched with the benchmark reference sets of protein complexes CYC2008 [40] and Complex Portal (CP) [41]. The former contains 408 known protein complexes while the latter contains 628 complexes. Two complexes match if their overlap score (OS) is larger than a certain threshold ω. For the current experiments, and based on many previous studies [21][42], this threshold is set to 0.2. The overlap score $OS(c,b)$ [14] is given by Equation (11).

Let the set of predicted complexes be represented by *C*, and the set of reference complexes by *B*. Precision (P), Recall (R) and F-measure (F) [43] are obtained through:

$$P = \frac{|\{c | c \in C \wedge \exists b \in B, c \text{ matches } b\}|}{|C|} \tag{13}$$

$$R = \frac{|\{b | b \in B \wedge \exists c \in C, b \text{ matches } c\}|}{|B|} \tag{14}$$

$$F = \frac{2 \times P \times R}{P + R} \tag{15}$$

Precision measures the number of predicted protein complexes that match at least one reference complex. Recall measures the degree at which a predicted protein complex matches a reference complex. F-measure combines these two metrics. Following most of the previous studies [21], the best sets of parameters for our proposed method are chosen using the F-measure.

A confusion matrix *T* with cell t_{ij} represents the number of proteins shared between *i*th reference complex and the *j*th predicted identified complex. Given *n* reference protein complexes and *m* predicted protein complexes. Sensitivity (Sn), Positive Predictive Value (PPV), and Accuracy (Acc) [44] are obtained through:

Table 2
Performance comparison with other protein complex prediction methods on the Gavin dataset.

Reference Dataset	Method's Name	Number of Complexes	P	R	F	Sn	PPV	ACC	MMR
CYC2008	MComplex	554	0.491	**0.451**	0.470	**0.484**	0.581	**0.529**	**0.229**
	DEC	337	0.614	0.424	**0.501**	0.394	**0.629**	0.498	0.223
	CSO	174	0.673	0.228	0.340	0.280	0.622	0.418	0.119
	ClusterONE	241	0.498	0.324	0.392	0.460	0.596	0.524	0.151
	COACH	318	0.519	0.333	0.406	0.440	0.556	0.495	0.175
	CMC	120	0.608	0.218	0.321	0.371	0.606	0.474	0.110
	MCODE	66	0.758	0.150	0.250	0.277	0.513	0.377	0.072
	NOCD-GCN	271	**0.915**	0.297	0.448	0.478	0.386	0.430	0.119
CP	MComplex	554	0.502	**0.389**	0.438	**0.437**	0.455	0.446	**0.163**
	DEC	337	0.647	0.371	**0.472**	0.352	0.490	0.415	0.163
	CSO	174	0.701	0.205	0.318	0.250	**0.506**	0.356	0.087
	ClusterOne	241	0.539	0.282	0.370	0.414	0.484	**0.448**	0.105
	COACH	318	0.579	0.290	0.386	0.393	0.455	0.423	0.117
	CMC	120	0.642	0.182	0.283	0.327	0.486	0.398	0.074
	MCODE	66	0.712	0.111	0.193	0.238	0.405	0.311	0.045
	NOCD-GCN	271	**0.926**	0.232	0.372	0.398	0.301	0.346	0.081

Notes: Number highlighted in bold represent the highest value for a certain metric. Underlined numbers represent the second highest value for a certain metric. Columns 4 to 10 refer to: P is Precision, R is Recall, F is F-measure, Sn is Sensitivity, PPV is Positive Predicted value, ACC is Accuracy and MMR is Maximum Matching Ratio.

$$Sn = \frac{\sum_{i=1}^{n} \max_{j} t_{ij}}{\sum_{i=1}^{n} N_i} \tag{16}$$

$$PPV = \frac{\sum_{j=1}^{m} \max_{i} t_{ij}}{\sum_{j=1}^{m} \sum_{i=1}^{n} t_{ij}} \tag{17}$$

$$Acc = \sqrt{Sn \times PPV} \tag{18}$$

where N_i represents the size of the ith reference protein complex. Sensitivity represents the number of shared proteins between matched complexes with respect to the size of the reference protein complexes. Positive Predictive Value represents the same concept with respect to the size of the predicted protein complexes. Both metrics study how the proteins are spread across the predicted complexes. However, the former metric can be inflated by putting all proteins in the same cluster while the latter metric can be inflated by putting every protein in its own cluster. Subsequently, accuracy is the geometric mean between the two measures to give a reasonable trade off as both of them need to be relatively high to obtain a high accuracy value.

An additional evaluation metric is the maximum matching ratio (MMR) [11]. It is used to negate the effect of the PPV measure as it is predisposed to decrease for clustering approaches that allow overlap between clusters. Given a matrix O of size *nxm* where *n* is the number of predicted complexes and *m* is the number of reference complexes, then o_{ij} is the overlap score between identified complex *i* and reference complex *j* given by Equation (11). This means that MMR ascertains the quality of the predicted complexes and their similarity in structure to the known complexes. MMR is given by:

$$MMR = \frac{\sum_{j=1}^{m} \max_{i=1}^{n} o_{ij}}{m} \tag{19}$$

3. Results

In this section, we compare the proposed method, MComplex, with state-of-the-art protein complex prediction approaches. We applied these approaches with parameter settings that are based on their respective references and studies. These methods are CSO [17], ClusterONE [11], COACH [13], CMC [16], MCODE [14] and NOCD-GCN [19]. We also present a focused comparison between our method and DEC [21], which is a similar approach that uses a temporal network and clusters the PPI network accordingly. The results are shown in Tables 2, 3, 4 and 5.

As shown in Table 2 for the CYC2008 reference dataset, our proposed method achieves the highest recall of 0.451, the highest sensitivity of 0.484, the highest accuracy of 0.529 and the highest MMR of 0.229. It also achieves the second highest F-measure. DEC achieves the highest F-measure of 0.501, the highest PPV of 0.629 and the second highest MMR of 0.223. Its relatively high recall and precision are the contributors to the elevated F-measure. The second highest sensitivity value of 0.478 is obtained by NOCD-GCN which uses a GCN to obtain a cluster affiliation matrix. It also obtains the highest precision with a value of 0.915 while MCODE obtains the second highest precision with 0.818 which can be attributed to it obtaining the lowest number of clusters. Furthermore, our method and DEC obtain the highest number of complexes, 554 and 337 respectively. As for the CP reference dataset, the same observation can be made regarding the precision, recall, F-measure and MMR in terms of the top two performing methods. The highest sensitivity value of 0.437 is obtained by MComplex which is similar to the results seen for CYC2008, and ClusterOne achieves the second highest with 0.414. Our approach obtains the second highest accuracy of 0.446 which is extremely close to the highest value of 0.448 obtained by ClusterOne. Additionally CSO achieves the highest PPV of 0.506 while DEC obtains the second highest with 0.490.

Results for the Krogan dataset with CYC2008 as the reference dataset, in Table 3, show a similar trend as Gavin dataset. Our proposed method obtains the highest recall of 0.608, the highest sensitivity score of 0.548, the highest accuracy of 0.574 and the highest MMR of 0.331, which is significantly higher than other methods. DEC obtains the highest F-measure of 0.489 due to relatively high precision and recall values, the highest PPV value of 0.69, as well as obtaining the second highest values for PPV, accuracy and MMR. In contrast, ClusterONE achieves the second highest PPV value of 0.681, and MCODE also obtain the lowest number of clusters which leads to high precision values. Meanwhile, CSO obtains the highest precision of 0.872. Furthermore, NOCD-GCN ends up obtaining the second highest values for precison, F-measure and sensitivity. The same observations can be seen when CP is used as a reference dataset with the changes being that MComplex obtains the second highest values for F-measure and MMR while DEC obtains the highest value for the latter. However, the values achieved for MMR are extremely similar with our approach obtaining 0.252 and DEC obtaining 0.256.

Regarding the third dataset MIPS and the CYC2008 reference dataset, our proposed method achieves the highest recall of 0.586 and

Table 3
Performance comparison with other protein complex prediction methods on the Krogan dataset.

Reference Dataset	Method's Name	Number of Complexes	P	R	F	Sn	PPV	ACC	MMR
CYC2008	MComplex	1215	0.376	**0.608**	0.465	**0.548**	0.601	**0.574**	**0.331**
	DEC	726	0.419	0.588	**0.489**	0.433	**0.690**	0.546	0.227
	CSO	141	**0.872**	0.211	0.340	0.263	0.625	0.405	0.114
	ClusterONE	240	0.579	0.328	0.419	0.399	0.682	0.521	0.189
	COACH	352	0.622	0.348	0.446	0.429	0.562	0.491	0.204
	CMC	111	0.748	0.235	0.358	0.381	0.589	0.474	0.123
	MCODE	76	0.75	0.169	0.276	0.257	0.577	0.385	0.084
	NOCD-GCN	271	0.785	0.348	0.482	0.478	0.386	0.430	0.148
CP	MCOMPLEX	1215	0.414	**0.541**	0.469	**0.489**	0.480	**0.484**	0.252
	DEC	726	0.486	0.529	**0.507**	0.387	**0.544**	0.459	**0.256**
	CSO	141	**0.901**	0.188	0.311	0.235	0.456	0.327	0.083
	ClusterOne	240	0.608	0.306	0.407	0.371	0.509	0.434	0.134
	COACH	352	0.656	0.326	0.436	0.387	0.403	0.395	0.154
	CMC	111	0.784	0.221	0.345	0.344	0.424	0.382	0.086
	MCODE	76	0.789	0.137	0.233	0.224	0.445	0.316	0.056
	NOCD-GCN	271	0.781	0.266	0.397	0.448	0.256	0.339	0.096

Notes: Number highlighted in bold represent the highest value for a certain metric. Underlined numbers represent the second highest value for a certain metric. Columns 4 to 10 refer to: P is Precision, R is Recall, F is F-measure, Sn is Sensitivity, PPV is Positive Predicted value, ACC is Accuracy and MMR is Maximum Matching Ratio.

Table 4
Performance comparison with other protein complex prediction methods on the MIPS dataset.

Reference Dataset	Method's Name	Number of Complexes	P	R	F	Sn	PPV	ACC	MMR
CYC2008	MComplex	1364	0.223	**0.586**	0.323	0.390	0.561	0.468	**0.243**
	DEC	1057	0.246	0.561	0.342	0.369	0.620	**0.479**	0.174
	CSO	111	**0.577**	0.162	0.253	0.171	0.642	0.331	0.074
	ClusterONE	259	0.371	0.245	0.295	0.246	**0.668**	0.406	0.102
	COACH	455	0.345	0.306	0.325	0.330	0.388	0.358	0.142
	CMC	139	0.568	0.206	0.302	0.254	0.491	0.353	0.084
	MCODE	54	0.555	0.096	0.163	0.188	0.412	0.278	0.088
	NOCD-GCN	294	0.517	0.309	**0.387**	**0.497**	0.257	0.357	0.107
CP	MCOMPLEX	1364	0.275	**0.567**	0.370	0.393	0.413	0.403	0.204
	DEC	1057	0.313	0.554	**0.400**	0.371	0.462	**0.414**	**0.218**
	CSO	111	0.448	0.178	0.255	0.208	0.357	0.273	0.053
	ClusterOne	259	0.417	0.247	0.310	0.251	0.468	0.343	0.079
	COACH	455	0.365	0.314	0.337	0.343	0.272	0.306	0.106
	CMC	139	**0.551**	0.167	0.256	0.178	**0.619**	0.332	0.040
	MCODE	54	0.537	0.070	0.124	0.210	0.256	0.232	0.026
	NOCD-GCN	294	0.514	0.228	0.316	**0.427**	0.207	0.297	0.068

Notes: Number highlighted in bold represent the highest value for a certain metric. Underlined numbers represent the second highest value for a certain metric. Columns 4 to 10 refer to: P is Precision, R is Recall, F is F-measure, Sn is Sensitivity, PPV is Positive Predicted value, ACC is Accuracy and MMR is Maximum Matching Ratio.

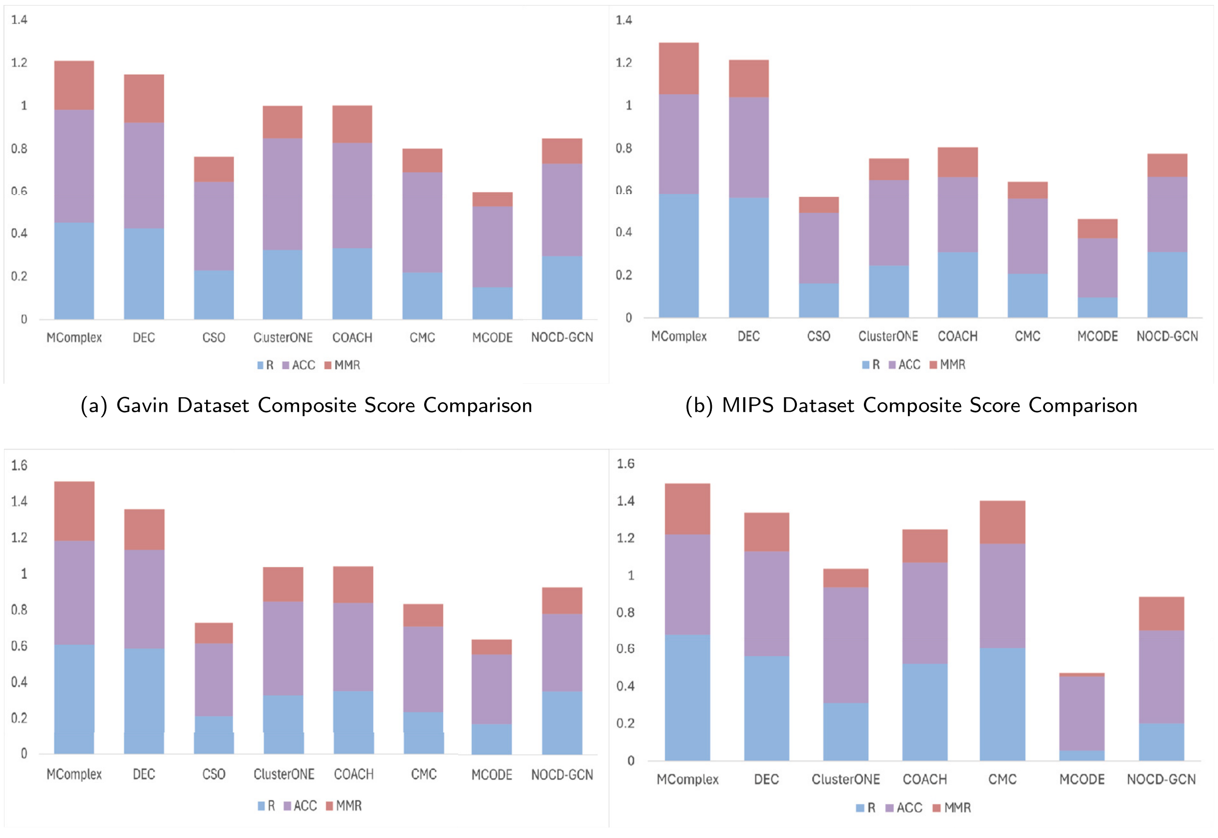
the highest MMR of 0.243. It also obtains the second highest value for accuracy with 0.468, while the highest value of 0.479 is obtained by DEC. The results are shown in Table 4. NOCD-GCN obtains the highest values of 0.387 and 0.497 for F-measure and sensitivity, respectively. Meanwhile CSO scores the highest precision value of 0.750. For PPV, the highest value of 0.668 is recorded by ClusterONE. Following this comparison, we can observe that the first three datasets almost have the same ranking for our proposed approach in regard to the various metrics, with MComplex obtaining the highest values for recall and MMR values, while also achieving the highest accuracy and sensitivity on the Gavin dataset and Krogan dataset, and the second highest accuracy on the MIPS dataset. The same observations regarding recall, sensitivity and accuracy are observed when the predicted complexes are compared to the known complexes of CP. Furthermore, when considering this reference dataset, CMC obtains the highest precision of 0.551 and highest PPV of 0.619. Meanwhile MCODE obtains the second highest precision of 0.537, and ClusterOne obtains the second highest PPV of 0.468. Our approach obtains the second highest values of 0.370 and 0.204 for F-measure and MMR respectively while DEC achieves the highest values for both metrics with 0.400 and 0.218, respectively.

The approaches are then tested on the STRING dataset which is more complex and thus more difficult to analyze due to its high number of interactions and proteins and its high level of noise. However, due to the heavy computations required to extract complexes from the STRING dataset, CSO cannot be used as it is a clique mining algorithm including additional gene ontology (GO) information, and thus, is not feasible in such a large and dense network. In contrast, our proposed method, as DEC [21], constructs a temporal PPI network with each subnetwork being sparser and smaller than the static PPI network. That is done before integration as only active proteins and active interactions are present in each one. Furthermore, given that each subnetwork undergoes clustering alone, this makes the two methods more suitable to handle very large networks as less computational time is needed to obtain a result. In particular, when considering the CYC2008 reference dataset, our proposed method obtains the highest recall of 0.684, the highest PPV of 0.542 and the highest MMR of 0.274 while DEC obtains the second highest value for accuracy. Meanwhile, NOCD-GCN obtains the highest precision of 0.747, ClusterONE obtains the highest accuracy of 0.629, CMC obtains the highest F-measure of 0.388, and COACH obtains the highest sensitivity of 0.956. The same observations can be made for the CP reference dataset.

Table 5
Performance comparison with other protein complex prediction methods on the STRING dataset.

Reference Dataset	Method's Name	Number of Complexes	P	R	F	Sn	PPV	ACC	MMR
CYC2008	MComplex	2738	0.188	**0.684**	0.295	0.531	**0.542**	0.537	**0.274**
	DEC	1260	0.212	0.561	0.308	0.842	0.388	0.571	0.208
	ClusterONE	837	0.146	0.309	0.198	0.843	0.469	**0.629**	0.099
	COACH	1548	0.244	0.522	0.333	**0.956**	0.315	0.548	0.177
	CMC	1109	0.286	0.603	**0.388**	0.919	0.354	0.570	0.231
	MCODE	130	0.146	0.054	0.079	0.643	0.247	0.398	0.019
	NOCD-GCN	158	**0.747**	0.200	0.317	0.721	0.357	0.507	0.178
CP	MCOMPLEX	2738	0.227	**0.637**	0.335	0.535	**0.437**	0.483	**0.232**
	DEC	1260	0.275	0.525	0.361	0.815	0.326	0.515	0.180
	ClusterOne	837	0.203	0.326	0.250	0.847	0.389	**0.574**	0.094
	COACH	1548	0.304	0.494	0.376	**0.936**	0.300	0.530	0.161
	CMC	1109	0.339	0.583	**0.429**	0.888	0.292	0.509	0.194
	MCODE	130	0.200	0.061	0.093	0.638	0.180	0.339	0.019
	NOCD-GCN	158	**0.703**	0.170	0.274	0.599	0.305	0.428	0.054

Notes: Number highlighted in bold represent the highest value for a certain metric. Underlined numbers represent the second highest value for a certain metric. Columns 4 to 10 refer to: P is Precision, R is Recall, F is F-measure, Sn is Sensitivity, PPV is Positive Predicted value, ACC is Accuracy and MMR is Maximum Matching Ratio.



(a) Gavin Dataset Composite Score Comparison

(b) MIPS Dataset Composite Score Comparison

(c) Krogan Dataset Composite Score Comparison

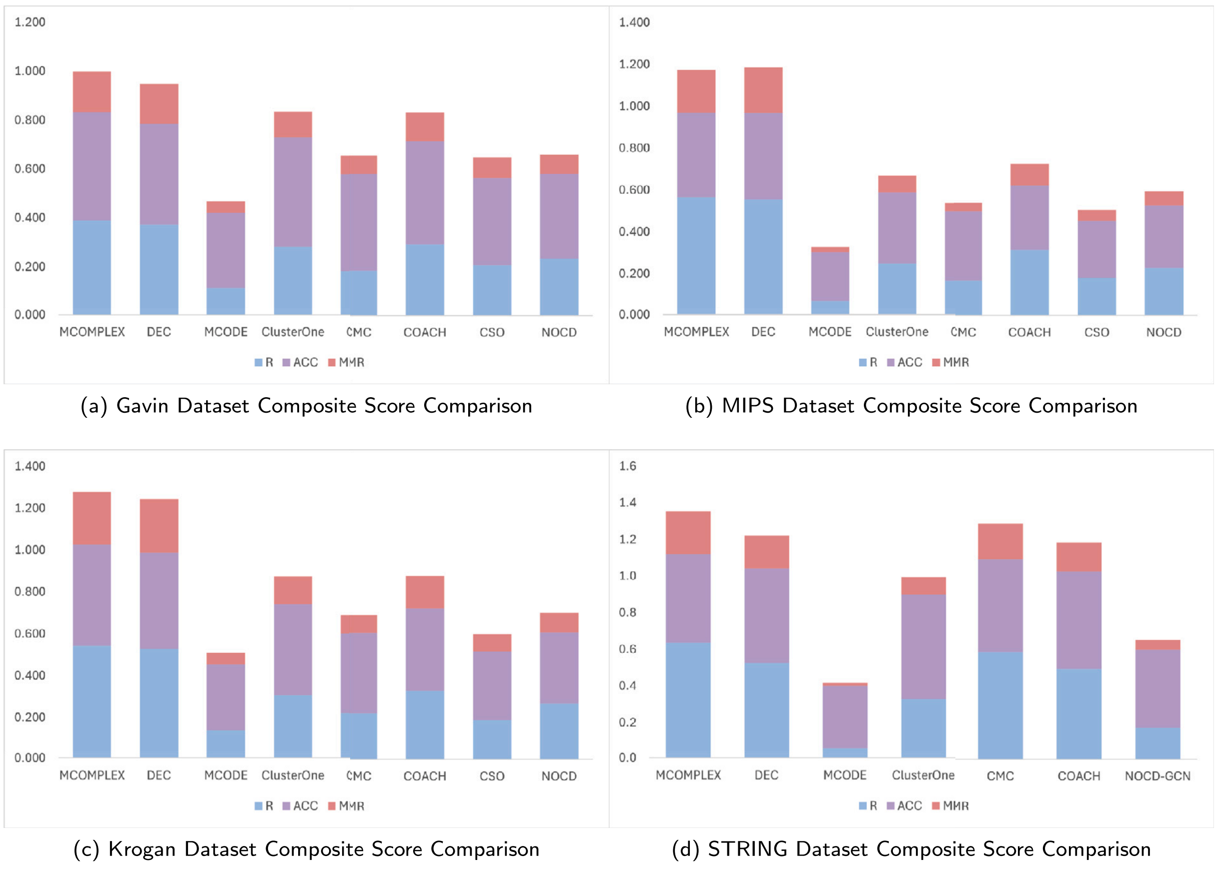
(d) STRING Dataset Composite Score Comparison

Fig. 6. Composite Score comparison using multiple approaches on different datasets with the CYC2008 as the reference complexes dataset: (a) Gavin Dataset, (b) MIPS Dataset, (c) Krogan Dataset and (d) STRING Dataset.

We also compare the various approaches across the four datasets using a composite score [11]. It is simply the sum of MMR, Recall, and Accuracy values.

The scores obtained on the four datasets with CYC2008 as the reference dataset are visualized in Fig. 6. It can be seen that our proposed approach obtains the highest composite score across all of the datasets with recall and MMR indicating the number and the quality of matched predicted complexes with reference ones. DEC obtains the second highest composite score for the three benchmark datasets while CMC obtains the second highest value for the STRING dataset. As shown in Fig. 7, the same observations hold true when CP is used as the reference dataset with a difference which is that DEC obtains a slightly higher score but extremely similar values to MComplex when checking the MIPS dataset.

In order to calculate the statistical significance of the obtained results and improvements achieved by MComplex, we apply the Friedman statistical test with a significance level of $\alpha = 0.05$. The null hypothesis, H_0, is defined as follows: *For a given performance metric PM, the results and their variations across different datasets are not significant.* Rejecting H_0 indicates that the differences observed in the metric *PM* across the models are statistically significant. The *p*-value and the conclusions of the sta-



(a) Gavin Dataset Composite Score Comparison

(b) MIPS Dataset Composite Score Comparison

(c) Krogan Dataset Composite Score Comparison

(d) STRING Dataset Composite Score Comparison

Fig. 7. Composite Score comparison using multiple approaches on different datasets with the Complex Portal as the reference complexes dataset: (a) Gavin Dataset, (b) MIPS Dataset, (c) Krogan Dataset and (d) STRING Dataset.

Table 6
Friedman statistical test at a significance level $\alpha = 0.05$.

Evaluation Metric	CYC2008		CP	
	p-value	Conclusion	*p*-value	Conclusion
P	0.017	Reject H_0	0.028	Rreject H_0
R	0.003	Reject H_0	0.002	Reject H_0
F	0.011	Reject H_0	0.007	Reject H_0
Sn	0.094	Fail to Reject H_0	0.186	Fail to Reject H_0
PPV	0.005	Reject H_0	0.017	Reject H_0
ACC	0.003	Reject H_0	0.004	Reject H_0
MMR	0.004	Reject H_0	0.003	Reject H_0
CS	0.002	Reject H_0	0.003	Reject H_0

Notes: The evaluation Metrics are the following: P is Precision, R is Recall, F is F-measure, Sn is Sensitivity, PPV is Positive Predicted value, ACC is Accuracy and MMR is Maximum Matching Ratio.

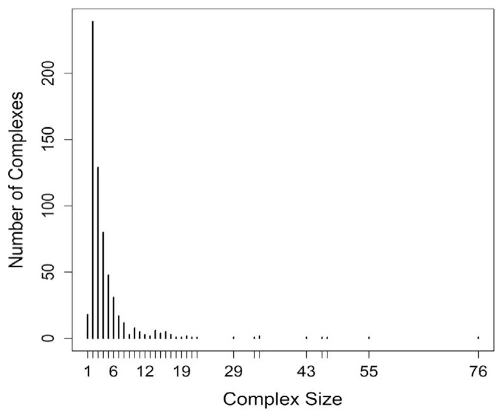
tistical test based on both reference protein complex datasets are shown in Table 6. We can observe that all evaluation metrics values are statistically significant except for sensitivity. Based on its definition, sensitivity measures the number of shared proteins between predicted and reference complexes with respect to the size of reference protein complexes. We note that MComplex scores the lowest sensitivity in both CYC2008 and CP reference datasets only for STRING dataset, while it scores the highest or the second highest values in the other four PPI datasets. As a result, the statistical significance across all four datasets is affected by the relatively low sensitivity values of STRING. We further discuss these results in the next section.

4. Discussion

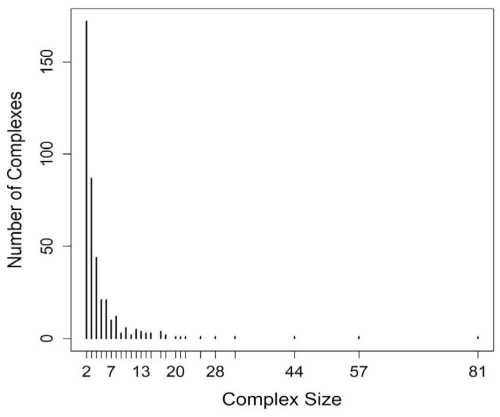
The experimental results prove that our proposed method works well with a dynamic network and even outperforms many state-of-the-art approaches over multiple datasets and metrics. In particular, given the recall values obtained and its statistical significance, it succeeds in generating clusters that match with the largest number of known complexes among the compared methods when taking into consideration both reference datasets. Furthermore, available proteins in a dataset do not get clustered in only a few identified complexes. While the results are not statistically significant for the sensitivity metric, the previous observation still stands as our approach obtains the highest sensitivity and accuracy on multiple datasets, with the sensitivity metric representing the number of proteins shared by matched pairs of identified and reference complexes with the highest overlap in regard to the sum of the number of proteins present in the benchmark.

Additionally, our approach can predict large complexes as well. Figs. 8 and 9 show the distribution of complex sizes across reference and predicted complex datasets, respectively. By comparing these distributions, we note that MComplex is able to predict complexes of various sizes with a distribution that matches the distribution of complex sizes in the reference datasets. Given that MComplex makes use of the Mapper algorithm, the latter does not limit the sizes of the predicted protein complexes, but it could be affected by the number of intervals, the percentage overlap and the clustering algorithm chosen. In our case, given the choice of Label Propagation, we can observe that the sizes of the predicted protein complexes are an additional indicator of the improved performance of MComplex. Furthermore, to examine the ability of our method to predict complexes of different types and functions, we consider three predicted complexes: Arp2/3 protein complex which consists of a fully connected component, as shown in Fig. 10(a), as well as Transcription factor TFIIIC complex and TRAPP complex which consist of sparser components, as shown in Figs. 10(b) and 10(c).

We also note that the predicted TRAPP complex also contains a novel protein YGR143W added by MComplex. Such protein could be subject

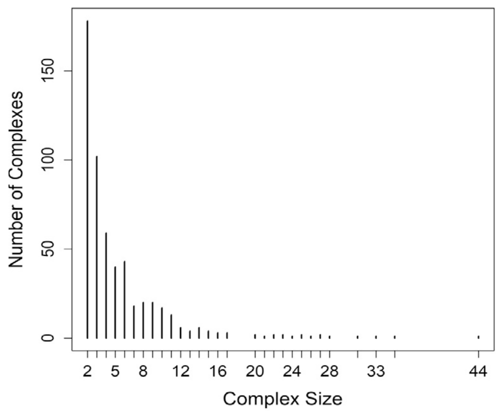


(a) CP Dataset

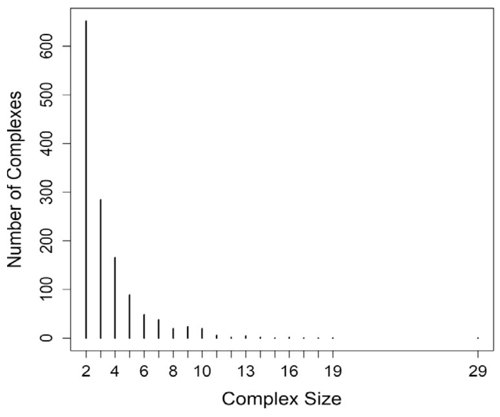


(b) CYC2008 Dataset

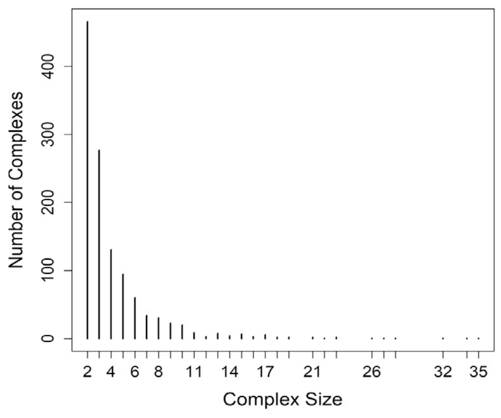
Fig. 8. Distribution of the sizes of protein complexes in the reference datasets: (a) CP Dataset, (b) CYC2008 Dataset.



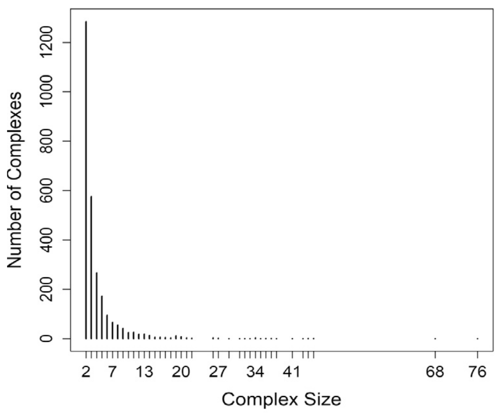
(a) Gavin Dataset



(b) MIPS Dataset



(c) Krogan Dataset

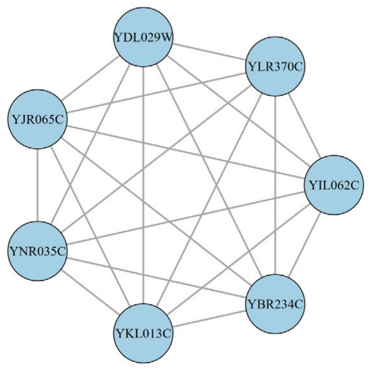


(d) STRING Dataset

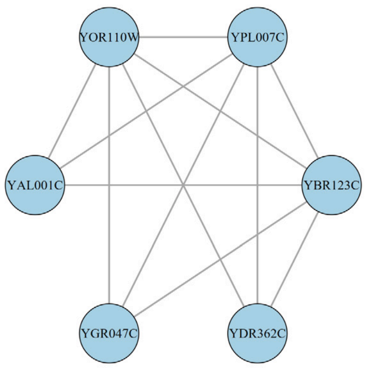
Fig. 9. Distribution of the sizes of predicted protein complexes by MComplex from benchmark datasets: (a) Gavin Dataset, (b) MIPS Dataset, (c) Krogan Dataset and (d) STRING Dataset.

to further study. In addition, based on the SGD's GO::TermFinder tool [45], all three complexes relate to different biological processes: nucleation of branched actin filaments for the Arp2/3 protein complex, regulation of transcription from RNA polymerase III (Pol III) promoters for the Transcription factor TFIIIC complex, and the transportation of vesicles from the ER through the Golgi to the plasma membrane for the TRAPP complex.

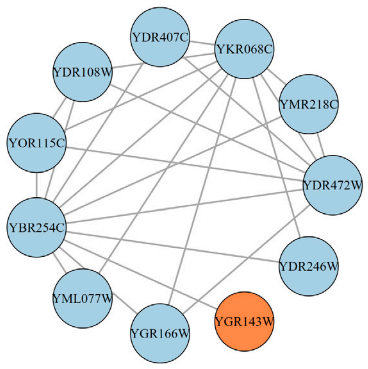
Furthermore, our approach can handle very large and noisy networks through the construction of a dynamic network which contains subnetworks that are smaller than the original static PPI network. Additionally, the Mapper algorithm segments the data contained in each subnetwork into smaller partitions. This further reduces the effect of the size of the input as well as its noise which is inherent to high-throughput data. Given that the algorithm utilizes topological concepts, such as dimen-



(a) Arp2/3 protein complex



(b) Transcription factor TFIIIC complex



(c) Predicted TRAPP complex

Fig. 10. Three protein complexes predicted by MComplex using the Gavin dataset and compared to the CYC2008 reference dataset. The blue nodes represent correctly identified proteins while the orange node represents an additional protein which could be subject to further study. The three complexes are: (a) Transcription factor TFIIIC complex, (b) Arp2/3 protein complex, and (c) TRAPP complex.

sionality reduction and data partitioning through a cover, it is resistant to such anomalies. This is also shown when MComplex is tested on the STRING dataset and its output matched the largest amount of known protein complexes. Subsequently, when taking into consideration the composite score, MComplex significantly outperforms the rest of the approaches examined in the experiments performed on the four datasets considered with the values being statistically significant. This further displays the quality of the predicted complexes in regard to known complexes as MMR, which is a component of the composite score and is also statistically relevant, represents the maximal overlap between the identified and reference complex sets. This is further supported as our approach obtains the highest values of this metric on all four datasets while considering both reference datasets except in two cases where the results are comparable to DEC. This means that our approach yields predicted protein complexes with a higher structure similarity to known protein complexes. They also match with the highest number of known protein complexes which is crucial since it means that the predicted complexes have a high probability of yielding true protein complexes. This also accelerates experimental work by narrowing down the search space usually done for experimental protein complex detection. In terms of limitations and potential improvements, we aim to reduce the overall number of predicted complexes. A possible way to do this would be by considering a smaller search space by including more information about proteins during the embedding stage. The latter could give better clustering results as proteins within the same complex typically share similar functions [15].

5. Conclusion

This work presents MComplex, an approach to predict protein complexes from high-throughput PPI datasets. Given that proteins interact with each other, such interactions are represented as a network where nodes and edges correspond to proteins and their interconnections, respectively. However, since protein interactions change based on different environmental conditions, a time-series gene expression dataset is integrated with the PPI network to model such dynamics. This generates a temporal network where each subnetwork represents active proteins and their interactions occurring at each time-step. Given that protein complexes are groups of interconnected proteins, we are interested in the neighborhood of each protein in the subnetworks of the temporal network. Hence, we use a GCN to obtain embeddings for each subnetwork based on patch representations of the surrounding region of each node. Since this work is an unsupervised learning approach, the GCN is trained using a GAN where it serves as an encoder that generates representations of both positive and negative examples. In this case, the positive example is the input network, while the negative example is a modified version of the same network where the feature matrix of the nodes undergoes row-wise shuffling. Each row of the feature matrix corresponds to a specific node's features values. Afterwards, the network embeddings obtained are then passed to the Mapper algorithm which uncovers underlying protein complexes.

Experimental results show that MComplex significantly outperforms state-of-the-art approaches based on different evaluation metrics especially concerning the similarity between predicted protein complexes and reference protein complexes that matched. Additionally, our proposed approach obtains the highest number of benchmark protein complexes matching at least one identified complex and the highest overlap percentage of these matches. This further proves the significance of our findings. Moreover, given the nature of the integrated data as a temporal network, MComplex is not overwhelmed by the size and denseness of a PPI network, as is the case in the STRING dataset. Following this, and through the generation of subnetwork representations, the Mapper algorithm divides each subnetwork into multiple segments which are then clustered individually. This further decreases the effect of the size of a network as well as the noise present in a dataset. However, since our proposed method relies on the Mapper algorithm to cluster the proteins, then it generates, by design, a high number of overlapping clusters. This leads to a higher number of communities than normal. This is resolved to a certain extent in the post-processing step. As future work, we plan to include information about the proteins themselves as features for the GCN to obtain a better lens function as proteins present in the same complex usually have the same functionality [15].

CRediT authorship contribution statement

Leonardo Daou: Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation, Formal analysis, Data curation. **Eileen Marie Hanna:** Writing – review & editing, Supervision, Project administration, Methodology, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.csbj.2024.10.009.

References

[1] Gu J, Bourne PE. Structural bioinformatics. Wiley-Blackwell; 2009.
[2] Spirin V, Mirny L. Protein complexes and functional modules in molecular networks. Proc Natl Acad Sci USA 2003;100:12123–8. https://doi.org/10.1073/pnas.2032324100.
[3] Zhang Q, Petrey D, Deng L, Qiang L, Shi Y, Thu C, et al. Structure-based prediction of protein-protein interactions on a genome-wide scale. Nature 2012;490:556–60. https://doi.org/10.1038/nature11503.
[4] De Las Rivas J, Fontanillo C. Protein–protein interactions essentials: key concepts to building and analyzing interactome networks. PLoS Comput Biol 2010;6:e1000807. https://doi.org/10.1371/journal.pcbi.1000807.
[5] Butland G, Peregrín-Alvarez J, Li J, Yang A, Yang X, Canadien V, et al. Interaction network containing conserved and essential protein complexes in escherichia coli. Nature 2005;433:531–7. https://doi.org/10.1038/nature03239.
[6] Chen K-C, Wang T-Y, Chan C-h. Associations between hiv and human pathways revealed by protein-protein interactions and correlated gene expression profiles. PLoS ONE 2012;7:1–10. https://doi.org/10.1371/journal.pone.0034240.
[7] Tzakos AG, Fokas D, Johannes C, Moussis V, Hatzimichael E, Briasoulis E. Targeting oncogenic protein-protein interactions by diversity oriented synthesis and combinatorial chemistry approaches. Molecules 2011;16:4408–27. https://api.semanticscholar.org/CorpusID:10345859.
[8] Acuner Ozbabacan SE, Keskin O, Nussinov R, Gursoy A. Enriching the human apoptosis pathway by predicting the structures of protein–protein complexes. J Struct Biol 2012;179:338–46. https://doi.org/10.1016/j.jsb.2012.02.002. https://www.sciencedirect.com/science/article/pii/S1047847712000342. Structural Bioinformatics.
[9] Gavin A-C, Bösche M, Krause R, Grandi P, Marzioch M, Bauer A, et al. Functional organization of the yeast proteome by systematic analysis of protein complexes. Nature 2002;415:141–7. https://api.semanticscholar.org/CorpusID:4425555.
[10] Uetz P, Giot L, Cagney G, Mansfield T, Judson R, Knight J, et al. A comprehensive analysis of protein-protein interactions in saccharomyces cerevisiae. Nature 2000;403:623–7. https://doi.org/10.1038/35001009.
[11] Nepusz T, Yu H, Paccanaro A. Detecting overlapping protein complexes in protein-protein interaction networks. Nat Methods 2012;9:471–2. https://doi.org/10.1038/nmeth.1938.
[12] Tadaka S, Kinoshita K. Ncmine: core-peripheral based functional module detection using near-clique mining. Bioinformatics 2016;32:btw488. https://doi.org/10.1093/bioinformatics/btw488.
[13] Wu M, li X, Kwoh C-K, Ng S-K. A core-attachment based method to detect protein complexes in ppi networks. BMC Bioinform 2009;10:169. https://doi.org/10.1186/1471-2105-10-169.
[14] Bader G, Hogue C. An automated method for finding molecular complexes in large protein interaction networks. BMC Bioinform 2003;4:2. https://doi.org/10.1186/1471-2105-4-2.
[15] Amin A, Shinbo Y, Mihara K, Kurokawa K, Kanaya S. Development and implementation of an algorithm for detection of protein complexes in large interaction networks. BMC Bioinform 2006;7:207. https://doi.org/10.1186/1471-2105-7-207.
[16] Liu G, Wong L, Chua HN. Complex discovery from weighted ppi networks. Bioinformatics (Oxford, England) 2009;25:1891–7. https://doi.org/10.1093/bioinformatics/btp311.
[17] Yijia Z, Lin H, Yang Z, Wang J, Li Y, Xu B. Protein complex prediction in large ontology attributed protein-protein interaction networks. IEEE/ACM Trans Comput Biol Bioinform 2013;10:729–41. https://doi.org/10.1109/TCBB.2013.86.
[18] Adamcsek B, Palla G, Farkas I, Derényi I, Vicsek T. Cfinder: locating cliques and overlapping modules in biological networks. Bioinformatics (Oxford, England) 2006;22:1021–3. https://doi.org/10.1093/bioinformatics/btl039.
[19] Zaki N, Singh H, Mohamed E. Identifying protein complexes in protein-protein interaction data using graph convolutional network. IEEE Access 2022;9:123717–26. https://doi.org/10.1109/ACCESS.2021.3110845.
[20] Wang J, Peng X, Li M, Pan Y. Construction and application of dynamic protein interaction network based on time course gene expression data. Proteomics 2013;13:301–12. https://doi.org/10.1002/pmic.201200277.
[21] Yijia Z, Lin H, Yang Z, Wang J, Liu Y, Sang S. A method for predicting protein complex in dynamic ppi networks. BMC Bioinform 2016;17. https://doi.org/10.1186/s12859-016-1101-y.
[22] Xie D, Yi Y, Zhou J, Li X, Wu H. A novel temporal protein complexes identification framework based on density–distance and heuristic algorithm. Neural Comput Appl 2019;31. https://doi.org/10.1007/s00521-018-3660-5.
[23] Xu Y, Zhou J, Zhou S, Guan J. Cpredictor3.0: detecting protein complexes from ppi networks with expression data and functional annotations. BMC Syst Biol 2017;11. https://doi.org/10.1186/s12918-017-0504-3.
[24] Veličković P, Fedus W, Hamilton WL, Liò P, Bengio Y, Hjelm RD. Deep graph infomax. In: International conference on learning representations; 2019. https://openreview.net/forum?id=rklz9iAcKQ.
[25] Rosen P, Hajij M, Wang B. Homology-preserving multi-scale graph skeletonization using mapper on graphs. arXiv, 2023.
[26] Bodnar C, Cangea C, Lio P. Deep graph mapper: seeing graphs through the neural lens. Front Big Data 2021;4:680535. https://doi.org/10.3389/fdata.2021.680535.
[27] Gavin A-C, Aloy P, Grandi P, Krause R, Boesche M, Marzioch M, et al. Proteome survey reveals modularity of the yeast cell machinery. Nature 2006;440:631–6. https://doi.org/10.1038/nature04532.
[28] Krogan N, Cagney G, Yu H, Zhong G, Guo X, Ignatchenko A, et al. Global landscape of protein complexes in the yeast saccharomyces cerevisiae. Nature 2006;440:637–43. https://doi.org/10.1038/nature04670.
[29] Güldener U, Münsterkötter M, Oesterheld M, Pagel P, Ruepp A, Mewes H-W, et al. Mpact: the mips protein interaction resource on yeast. Nucleic Acids Res 2006;34:D436–41. https://doi.org/10.1093/nar/gkj003.
[30] Franceschini A, Szklarczyk D, Frankild S, Kuhn M, Simonovic M, Roth A, et al. String v9.1: protein-protein interaction networks, with increased coverage and integration. Nucleic Acids Res 2012;41. https://doi.org/10.1093/nar/gks1094.
[31] Tu B, Kudlicki A, Rowicka M, Mcknight S. Logic of the yeast metabolic cycle: temporal compartmentalization of cellular processes. Science (NY) 2005;310:1152–8. https://doi.org/10.1126/science.1120499.
[32] Clough E, Barrett T. The gene expression omnibus database 2016;1418:93–110. https://doi.org/10.1007/978-1-4939-3578-9_5.
[33] Noori S, Al-a'araji N, Al-Shamery E. Construction of dynamic protein interaction network based on gene expression data and quartile one principle. Proteins, Struct Funct Bioinform 2022;90. https://doi.org/10.1002/prot.26304.
[34] Zhang Y, Du N, Li K, Feng J, Jia K, Zhang A. msidbn: a method of identifying critical proteins in dynamic ppi networks. BioMed Res Int 2014;2014. https://doi.org/10.1155/2014/138410.
[35] Loughrey C, Fitzpatrick P, Orr N, Jurek-Loughrey A. The topology of data: opportunities for cancer research. Bioinformatics 2021;37. https://doi.org/10.1093/bioinformatics/btab553.
[36] Clevert D-A, Unterthiner T, Hochreiter S. Fast and accurate deep network learning by exponential linear units (ELUs). In: Bengio Y, LeCun Y, editors. 4th international conference on learning representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference track proceedings; 2016.
[37] McInnes L, Healy J. Umap: uniform manifold approximation and projection for dimension reduction. J Open Sour Softw 2018.
[38] Raghavan N, Albert R, Kumara S. Near linear time algorithm to detect community structures in large-scale networks. Phys Rev E, Stat Nonlinear Soft Matter Phys 2007;76:036106. https://doi.org/10.1103/PhysRevE.76.036106.

[39] Hanna EM, Zaki N. Detecting protein complexes in protein interaction networks using a ranking algorithm with a refined merging procedure. BMC Bioinform 2014;15. https://doi.org/10.1186/1471-2105-15-204.

[40] Pu S, Wong J, Turner B, Cho E, Wodak S. Up-to-date catalogues of yeast protein complexes. Nucleic Acids Res 2009;37:825–31. https://doi.org/10.1093/nar/gkn1005.

[41] Meldal BHM, Perfetto L, Combe C, Lubiana T, Ferreira Cavalcante JaV, Bye-A-Jee H, et al. Complex portal 2022: new curation frontiers. Nucleic Acids Res 2021;50:D578–86. https://doi.org/10.1093/nar/gkab991.

[42] li X, Wu M, Kwoh C-K, Ng S-K. Computational approaches for detecting protein complexes from protein interaction networks: a survey. BMC Genomics 2010;11(Suppl 1):S3. https://doi.org/10.1186/1471-2164-11-S1-S3.

[43] Chua HN, Ning K, Sung W-K, Leong H, Wong L. Using indirect protein-protein interactions in protein complex prediction. Computational Systems Bioinformatics 2007;6:97–109. https://doi.org/10.1142/9781860948732_0014.

[44] Brohée S, van Helden J. Evaluation of clustering algorithms for protein interaction networks. BMC Bioinform 2006;7:488. https://doi.org/10.1186/1471-2105-7-488.

[45] Boyle EI, Weng S, Gollub J, Jin H, Botstein D, Cherry JM, et al. Termfinder–open source software for accessing gene ontology information and finding significantly enriched gene ontology terms associated with a list of genes. Bioinformatics 2004;20(18):3710–5. https://api.semanticscholar.org/CorpusID:17620249.