BraneMF: Integration of Biological Networks for Functional Analysis of Proteins

Surabhi Jagtap[a,b], Abdulkadir Çelikkanat[c], Aurélie Pirayre[a], Frédérique Bidard[a], Laurent Duval[a], Fragkiskos D. Malliaros[b]

[a] *IFP Energies nouvelles, 1 et 4 avenue de Bois-Préau, 92852 Rueil-Malmaison, France*
[b] *Université Paris-Saclay, CentraleSupélec, Inria, Centre for Visual Computing, 91190 Gif-Sur-Yvette, France*
[c] *Technical University of Denmark, DTU Compute, 2800 Kongens Lyngby, Denmark*

Abstract

Motivation: The cellular system of a living organism is composed of interacting bio-molecules that control cellular processes at multiple levels. Their correspondences are represented by tightly regulated molecular networks. The increase of omics technologies has favored the generation of large-scale disparate data and the consequent demand for simultaneously using molecular and functional interaction networks: gene co-expression, protein–protein interaction (PPI), genetic interaction, and metabolic networks. They are rich sources of information at different molecular levels, and their effective integration is essential to understand cell functioning and their building blocks (proteins). Therefore, it is necessary to obtain informative representations of proteins and their proximity, that are not fully captured by features extracted directly from a single informational level. We propose BraneMF, a novel random walk-based matrix factorization method for learning node representation in a multilayer network, with application to omics data integration.

Results: We test BraneMF with PPI networks of *Saccharomyces cerevisiae*, a well-studied yeast model organism. We demonstrate the applicability of the learned features for essential multi-omics inference tasks: clustering, function and PPI prediction. We compare it to state-of-the-art integration methods for multilayer networks. BraneMF outperforms baseline methods by achieving high prediction scores for a variety of downstream tasks. The robustness of results is assessed by an extensive parameter sensitivity analysis.

Availability: BraneMF's code is freely available at: https://github.com/Surabhivj/BraneMF, along with datasets, embeddings, and result files.

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1. Introduction

Comprehensive interpretation of biological mechanisms in an organism requires understanding biomolecular interactions represented by tightly regulated molecular networks [40]. The advent of high-throughput technologies has introduced enormous disparate omics data in the scenario, and, in parallel, promising avenues are paved for their analysis and interpretation [47]. Despite the explosion of omics data, functional annotations are yet to be unveiled, for at least 20% of proteins, even in model organisms [45]. Hence, novel methods to analyse and obtain significant knowledge from heterogeneous data are necessary. Analysis of single omics is limited to correlations, mostly reflecting reactive processes rather than causative ones. Several studies have shown the

Email address: `fragkiskos.malliaros@centralesupelec.fr` (Fragkiskos D. Malliaros)

Accepted to Bioinformatics, Oxford University Press

importance of multi-omics integration over single omics analysis [40]. Such an approach can provide insights into the interconnectedness of different bio-molecules (proteins, RNAs, metabolites) and the flow of biological information occurring among them. This will enable us to understand biological mechanisms, for instance, gene regulation [18, 20], protein function prediction (PFP) [9], or drug–target identification [28].

In the past years, network approaches have offered potential for integrative omics analysis, facilitating a new era of systems biology [40, 46, 10]. Nevertheless, it is necessary to obtain informative representations (e.g., embeddings) for the nodes in the network (bio-molecules) and their proximity. Potentially, this would be possible by modeling biological data as a multilayer network and learning integrated embeddings that could effectively capture richer features and preserve biological information from each individual layer. Multilayer network integration strategies can be classified as early, intermediate or late integration [25, 14]. In early integration methods, datasets are combined into a single dataset on which the model is built and the features are learned. In the late network integration strategy, a model for each network is built individually, and these individual network features are then combined. In intermediate integration, the data is combined through a joint model inference. Indeed, there is a great value in developing efficient intermediate-level integration approaches [50], capable of handling heterogeneous data, providing insights into the functional categories of proteins (e.g., representation of system-level inter-relationships within bio-molecules).

In this paper, we are inspired by Graph Representation Learning (GRL) algorithms to encode graph structure into compact embedding vectors [17]. Our motivation is further extended towards leveraging closed forms of GRL methods that perform implicit matrix factorization, favouring intrinsic connection and interpretability of graph topology [24, 35]. We propose BraneMF, a novel intermediate-level network integration framework by effectively combining PPI networks of heterogeneous data sources. We formally define the task as a multilayer network embedding problem. Given a set of networks, we aim to learn low-dimensional latent node embeddings so that the structure of input networks is properly integrated and preserved in the new space.

More formally, a multilayer graph of L-layers is a set $\mathcal{G} = \{\mathcal{G}_l\}_{l=1}^{L} = \{(\mathcal{V}_l, \mathcal{E}_l)\}_{l=1}^{L}$ of graphs, where $\mathcal{V}_l := \{v_1, \ldots, v_{N_l}\}$ and $\mathcal{E}_l := \{e_{1_l}, \ldots, e_{M_l}\}$ are the vertex and the undirected edges sets respectively. N_l and M_l denote the number of nodes and edges for each layer. Throughout the paper, we assume that the layers share the same set of nodes, so $\mathcal{V}_j = \mathcal{V}_l = \mathcal{V}$, and $N_j = N_l = |\mathcal{V}|$ for every $1 \leq j < l \leq L$. We use $\mathbf{A}^{(l)}$ to denote the adjacency matrix of the associated layer $\mathcal{G}_l$. Our main goal is to learn a low-dimensional feature representation for all $|\mathcal{V}|$ nodes. This integrated d-dimensional representation of $\mathcal{G}$ is given by $\mathbf{\Omega}_d \in \mathbb{R}^{|\mathcal{V}| \times d}$ ($d \ll |\mathcal{V}|$). We employ these embeddings for different downstream prediction tasks dedicated to the functional analysis of proteins.

2. Related work

A plethora of GRL approaches are based on random walks [33, 15, 30, 3], matrix factorization [24, 35], or neural networks [17]. However, they have mostly been introduced for single-layer networks [47]. Inspired by Word2Vec-based single-layer network embedding techniques [29, 33, 15], a few GRL methods have been proposed for multilayer networks. Principled Multilayer Network Embedding (PMNE) [26] is an extension of a single-layer graph embedding to a multilayer network. Multiplex Network Embedding (MNE) [48] is a multi-layer network embedding method that generates random walks for each layer and then applies the Skip-Gram model [29] to learn joint embeddings for each node. The final node embeddings are composed of three parts: common embeddings, relation-based embeddings, and a transformation matrix. Multi-Net [2] uses random walks, namely classical, diffusive, and physical to obtain node sequences [38, 16]. Then, it merges the nodes' neighborhoods (context for each node) and learns a d-dimensional feature vector for each node by maximizing the likelihood of occurrence of node neighbors across all layers. Multi-node2vec (Multi-n2v) [44] extends Node2Vec [15] to multi-layer networks. The model collects a bag of words from each layer by performing vertex neighborhood search. Then, the optimization procedure computes the features by using the Skip-Gram neural network model on the identified neighborhoods. Recently, MultiVERSE [34] computes a similarity matrix using random walk with restart (RWR). Then, it applies an optimized version of VERSE [42], a vertex-to-vertex similarity-oriented embedding method to compute

the representations. FAME [27] decomposes the heterogeneous multiplex network into homogeneous and bipartite sub-networks. It follows the use of a spectral transformation module to automatically aggregate and decouple sub-networks with the exploration of their multi-relational topological signals. Lastly, BraneExp [20] is based on random walks that are used to define node similarity, using expressive conditional probability functions. It is a network integration framework with the concept of exponential family graph embeddings, which generalizes multilayer random walk-based GRL methods to an instance of exponential family distribution [36, 3].

For biological networks, Similarity Network Fusion (SNF) [43] constructs a similarity network for each data type and then iteratively integrates these networks using a network fusion methodology. Mashup [7] is a network integration framework based on matrix factorization that builds compact low-dimensional vector representations of proteins. It takes as input a collection of PPI networks and generates embeddings that best explain their wiring patterns across all networks. deepNF [13] is a network fusion method relying on multimodal deep autoencoders (MDA). It also takes a collection of PPI networks as input and makes use of an autoencoder (AE). OhmNet [52] is an unsupervised feature learning approach for multilayer networks with a predefined hierarchy describing relationships between the layers. To capture the structural properties of networks, deepNF and Mashup are based on Random Walks with Restarts (RWR). The vectors learned from both methods are then fed into a support vector machine (SVM) classifier to predict functional classes of proteins. deepMNE-CNN [32] has been introduced as a multi-network embedding approach which applies a semi-autoencoder based model to learn protein features. Graph2GO [12] extends variational graph autoencoders (VGAE) to multilayer networks. It establishes to integrate networks derived from heterogeneous information, including sequence similarity, protein-protein interaction, and protein features, amino acid sequence, sub-cellular location, and protein domains.

Most of these network integration frameworks are engrossed towards Gene Ontology (GO) prediction. If two proteins have a similar function, apart from their direct relationship in the network, they can have many further characteristics in common, such as biological processes, molecular function, cellular location, regulated by the same transcription factor, have the same epigenetic mark or belong to the same metabolic pathway. In order to determine such similarities between *a priori* unlinked proteins, it is necessary to obtain an informative representation of proteins and their proximity that is not fully captured by handcrafted features directly extracted from the PPI network. GRL-driven models are candidates for the above tasks. Given a multilayer network, GRL algorithms can embed it into a new compact vector space in such a way that both the original network structure and other latent features are captured. Indeed, existing methods are challenged when applied to biological datasets that demand comprehensive handling of data heterogeneity. Also, existing GRL methods for multilayer networks depend on numerous parameters—thus being computationally intensive in finding optimal parameter settings. Besides, biologists generally dispose of low levels of ground truth. To efficiently search for appropriate ground truth when biological information is not fully known, becomes a difficult and time consuming task. Hence, there is a huge scope to develop new methods that can address these challenges. In this study, we derive embeddings purely via a data-driven fashion such that the probability of the context of a protein is maximized. To do so, we obtain random walk-based Positive Pointwise Mutual Information (PPMI) matrices from the set of networks to capture node neighborhood information. These matrices are used as input for learning embeddings using a joint singular value decomposition (SVD) framework. Moreover, we demonstrate the adequacy of the learned embeddings to solve important downstream machine learning tasks such as protein clustering with their functional enrichment, prediction of protein functions and PPIs.

3. Materials and methods

BraneMF is an integration framework to learn protein features from multiple PPI networks. A schematic representation is given in Fig. 1.

3.1. Computation of random walk-based PPMI matrices

Network properties, particularly topological ones, can unravel important information about the graph structure. While handling multiple heterogeneous networks that correspond to diverse characteristics, it is essential to extract

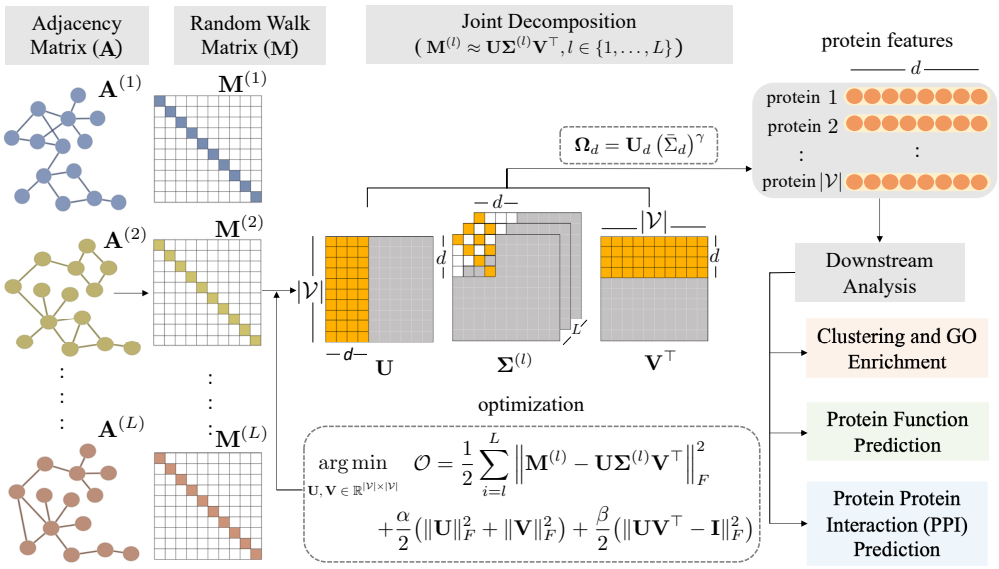


Figure 1: **Schematic representation of BraneMF.** The framework takes as input a set of PPI networks represented by their adjacency matrices $\mathbf{A}^{(l)}, l \in \{1, 2, \ldots, L\}$. For each PPI network, the random walk matrix $\mathbf{M}^{(l)}$ is computed. For integrative analysis, we learn protein features by jointly decomposing these random walk matrices $\mathbf{M}^{(l)}$ into $\mathbf{U}\mathbf{\Sigma}^{(l)}\mathbf{V}^\top$. The protein features $\mathbf{\Omega}_d$ are given by $\mathbf{U}_d(\bar{\mathbf{\Sigma}}_d)^\gamma$, where d is the embedding dimension and γ is a factor that scales the magnitude of the singular values. The learned protein features are then utilized for functional analysis of proteins.

relevant information concealed in their topology. We aim to extract such information from a multilayer graph $\mathcal{G}$, constructing a set of PPMI matrices that can delineate node similarity via random walks. Random walks, defined as node paths that consist of a series of random steps on the graph, have been used as a similarity measure for a variety of problems in graph theory. More precisely, from some single-layer graph $\mathcal{G}_l$ and a starting node v_i, a random walk performs transitions by selecting a neighborhood node at random at each step. The random walk generation procedure continues for all nodes and for a predefined number of walks of a given length. In this way, node sequences are obtained that are further used to learn node features. This can be achieved by maximizing the likelihood of node co-occurrences within random walks, following ideas from the Skip-Gram model in natural language processing (NLP). Nevertheless, for large networks, simulating random walks is computationally expensive and requires additional parameter settings. To alleviate this effect, recent studies have formulated the problem of random walk embedding generation as a matrix factorization task [24, 35].

Focusing on a specific instance of such approaches, the DeepWalk method first generates a corpus $\mathcal{W}$ by performing random walks on a graph [33]. A corpus $\mathcal{W}$ is a bag of multisets that counts the multiplicity of nodes v and their context c. DeepWalk then trains a Skip-Gram model on $\mathcal{W}$. To be formal, it assumes a corpus of node sequences represented as $v_1, v_2, \ldots, v_N$, where N is the length of the random walk. The context of node v_i is given as the surrounding nodes in a $2w$-sized window $\{v_{i-w}, \ldots, v_{i-1}, v_{i+1}, \ldots, v_{i+w}\}$, $w < N$. Following Levy and Goldberg [24] and Qiu et al. [35], the closed form expression of the DeepWalk matrix for any graph $\mathcal{G}_l$ is given by:

$$\underbrace{\log\left(\frac{\#(v,c)|\mathcal{W}|}{\#(v).\#(c)}\right) - \log b}_{\text{Skip-Gram}} = \underbrace{\log\left(\frac{\text{vol}(\mathcal{G}_l)}{bw}\left[\frac{1}{w}\sum_{r=1}^{w}\mathbf{P}^r\right]\mathbf{D}^{-1}\right)}_{\text{DeepWalk matrix}}.$$

On the left-hand side, $\#(v,c)$, $\#(v)$, and $\#(c)$ denote respectively the number of times node-context pair (v,c), node v and context c appear in $\mathcal{W}$, while b is the number of negative samples. The right-hand side is represented by $\mathbf{D}$ as the degree matrix of graph $\mathcal{G}_l$, and the power matrix

$\mathbf{P}$ defined as $\mathbf{D}^{-1}\mathbf{A}$. Here, vol($\mathcal{G}_l$) is the volume (size) of $\mathcal{G}_l$. For a detailed theoretical explanation, refer to Sec. 2.2 in [35]. Inspired by this formulation, we have chosen to obtain the set of PPMI matrices $\mathbf{M} = \{\mathbf{M}^{(l)}\}_{l=1}^{L}$ using the closed form expression of the DeepWalk matrices $\mathbf{M}^{(l)}$ for a multilayer graph $\mathcal{G}$:

$$\mathbf{M} = \left\{ \log\left(\frac{\text{vol}(\mathcal{G}_l)}{bw} \left[\frac{1}{w} \sum_{r=1}^{w} \left(\mathbf{P}^{(l)}\right)^r \right] \left(\mathbf{D}^{(l)}\right)^{-1} \right) \right\}_{l=1}^{L}. \quad (1)$$

Each matrix $\mathbf{M}^{(l)}$ corresponds to the DeepWalk matrix of $\mathcal{G}_l$ when the length of random walks goes to infinity. In this regard, $\mathbf{M}^{(l)}$ is different from the PPMI matrices computed in previous approaches. As discussed in Sec. 2, the PPMI matrix for deepNF and Mashup is computed using Random Walks with Restart (RWR), considering an additional parameter that controls the restart probability of the random walk. Despite both capturing node proximity, the DeepWalk matrix significantly differs from RWR; the formulation ensures that its latent factors will derive embeddings that capture node co-occurrences in random walks.

3.2. Joint representation learning for multilayer networks

The set of matrices $\mathbf{M}$ computed as above captures node proximity that still represents high-dimension protein features. As a consequence of the curse of dimensionality, these features are not compatible for downstream prediction tasks. Therefore, we want to obtain low-dimension integrated protein features that could be easily fed to any downstream machine learning tasks of interest. Nevertheless, our integration framework is developed on the construction of random walk-based PPMI matrices (Eq. (1)), on which joint matrix factorization is eventually performed. In order to learn the spectrum of one layer in graph $\mathcal{G}$, the singular values and singular vectors of its PPMI matrix $\bar{\mathbf{M}}$ can be obtained using SVD, as $\bar{\mathbf{M}} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U}$ and $\mathbf{V}$ correspond to the left and right singular vector matrices, and $\mathbf{\Sigma}$ is the diagonal singular value matrix. In the case of a multilayer graph $\mathcal{G}$ composed by L layers, we have L symmetric PPMI matrices. As a natural extension, we propose to approximate each PPMI matrix $\mathbf{M}^{(l)}$ by a set of jointly decomposed singular vector and singular value matrices shared by all layers, given by: $\mathbf{M}^{(l)} \approx \mathbf{U}\mathbf{\Sigma}^{(l)}\mathbf{V}^\top,\ l \in \{1, \ldots, L\}$. The correspondence above keeps, where $\mathbf{U}$ and $\mathbf{V}^\top$ are orthogonal matrices containing the joint singular vectors and $\mathbf{\Sigma}^{(l)} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ contains the corresponding singular values in the layer l. The minimization of the following objective function O yields $\mathbf{U}$ and $\mathbf{V}$:

$$\begin{aligned} \underset{\mathbf{U},\mathbf{V}\in\mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}}{\arg\min}\ O = \frac{1}{2}\sum_{l=1}^{L} \|\mathbf{M}^{(l)} - \mathbf{U}\mathbf{\Sigma}^{(l)}\mathbf{V}^\top\|_F^2 \\ + \frac{\alpha}{2}(\|\mathbf{U}\|_F^2 + \|\mathbf{V}\|_F^2) + \frac{\beta}{2}\|\mathbf{U}\mathbf{V}^\top - \mathbf{I}\|_F^2, \end{aligned} \quad (2)$$

where $\mathbf{I} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ is the identity matrix, and $\|\cdot\|_F$ denotes the Frobenius norm. The first term of the objective function O measures the overall approximation error when all layers are decomposed over $\mathbf{U}$. The second term, the norms of $\mathbf{U}$ and $\mathbf{V}^\top$, is added to improve numerical stability for the solutions; and the last term is a constraint to ensure that $\mathbf{V}^\top$ is close to the inverse of $\mathbf{U}$ ($\mathbf{M}^{(l)}$ is a symmetric matrix, thus its SVD can be given by $\mathbf{U}\mathbf{\Sigma}\mathbf{U}^{-1}$).

We solve the problem in Eq. (2) to get $\mathbf{U}$ and $\mathbf{V}^\top$. Since Eq. (2) is not jointly convex on $\mathbf{U}$ and $\mathbf{V}^\top$, we adopt an alternating scheme to find a local minimum for O by fixing $\mathbf{V}^\top$ first and optimizing on $\mathbf{U}$, and vice versa [11]. The derivatives of O with respect to $\mathbf{U}$ and $\mathbf{V}^\top$ are given in Eq. (S1) (Supplementary Material (SM)). For nonconvex optimization, a good initialization is important, and we suggest to compute the SVD of the mean for all matrices $\mathbf{M}^{(l)}$, and initialize $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{V}^\top$ with the resulting matrices: $\mathbf{U}$ is the set of joint singular vectors, namely a joint spectrum shared by all layers in $\mathcal{G}$, $\bar{\mathbf{\Sigma}}$ is the joint singular value matrix computed by taking the average of $\mathbf{\Sigma}^{(l)}$ matrices. The integrated embeddings $\mathbf{\Omega}_d \in \mathbb{R}^{n\times d}$ are obtained by multiplying the first d columns of $\mathbf{U}$, $\mathbf{U}_d \in \mathbb{R}^{n\times d}$, scaled by the γ-th power of the singular value magnitudes, $(\bar{\mathbf{\Sigma}}_d)^\gamma \in \mathbb{R}^{d\times d}$:

$$\mathbf{\Omega}_d = \mathbf{U}_d(\bar{\mathbf{\Sigma}}_d)^\gamma. \quad (3)$$

The stopping condition is defined by the convergence behavior of the cost function—the difference between its values for two consecutive iterations. This optimization process is similar to [11], that uses an eigendecomposition to find low-rank eigenvector matrices that are shared by all graph layers. However, these matrices were not random walk-based and the joint decomposition is performed differently. The joint SVD process described above is

essentially based on integrating information from multiple graph layers. It tends to treat each graph equitably, building a solution that smoothens out specificities of each layer. An overview of BraneMF is given in Algorithm 1 in the SM.

4. Results

4.1. Experimental setup

To substantiate our methodology, we apply it over six yeast STRING networks. Their relationships are mainly defined by 'Neighborhood', 'Fusion', 'Co-occurrence, 'Co-expression', 'Experimental', and 'Database'. A brief overview of the input data and the gene ontology (GO) terms are given in SM (Table S1) and SM (Table S2), respectively. Let $\mathcal{G}$ be a multilayer network constructed over $|\mathcal{V}| = 4,900$ proteins. BraneMF depends on three parameters: embedding size d, Eq. (3), window size w, Eq. (1), and weighting factor γ, Eq. (3). Given this set as input to BraneMF, d-dimensional latent features, $\mathbf{\Omega}_d \in \mathbb{R}^{|\mathcal{V}|\times d}$, are learned. We have selected nine baselines in our empirical analysis. Their description and parameter selection are provided in SM (Sec. 4). Further, we investigate the usefulness of the learned embeddings for various omics inference tasks. Firstly, we perform clustering using the protein features and by GO enrichment analysis we show the functional relatedness of proteins in these clusters (Sec. 4.2). Secondly, we design protein function prediction as a multi-label node classification problem by training a SVM model. We predict biological process (BP), molecular function (MF), and cellular component (CC) (Sec. 4.3). Next, we perform PPI prediction by learning protein features using datasets of 2015 (deepNF and Mashup benchmark datasets) aiming to predict unseen PPIs which are updated by 2021 [41]. Moreover, we use the computed embeddings from our dataset to reconstruct the integrated yeast PPI network provided in STRING database (Sec. 4.4). Note that, for each d, we select one embedding file per model that provides the best performance for each downstream task. The respective parameters selection is given in SM (Tables S4; S5). Lastly, we have also performed parameter uncertainty analysis, examining the impact of the different choices for d, γ, and w. Details are provided in SM (Sec. 8). The experiments and their respective evaluations are described below.

Method	$k = 40$	$k = 60$	$k = 80$	$k = 100$
SNF	7.2 ± 10.2	23.1 ± 6.6	15.2 ± 2.4	43.1 ± 4.1
Mashup	21.5 ± 0.6	30.1 ± 3.0	38.8 ± 0.4	41.9 ± 0.4
deepNF	**25.7 ± 1.7**	22.7 ± 1.2	26.3 ± 1.0	26.2 ± 1.1
Multi-Net	20.4 ± 2.2	22.5 ± 0.5	45.4 ± 0.0	45.4 ± 0.0
Multi-n2v	16.6 ± 2.4	24.7 ± 0.9	46.6 ± 0.1	46.6 ± 0.1
OhmNet	15.1 ± 7.2	35.1 ± 0.2	45.1 ± 0.3	45.1 ± 0.4
MultiVERSE	16.4 ± 10.4	13.7 ± 0.9	20.1 ± 0.0	20.4 ± 0.0
BraneExp	21.0 ± 6.7	41.9 ± 1.4	45.4 ± 0.0	45.4 ± 0.0
Graph2GO	21.3 ± 1.7	22.5 ± 11.9	25.5 ± 11.9	30.4 ± 7.7
BraneMF	24.05 ± 9.3	**46.2 ± 5.5**	**48.9 ± 4.28**	**49.5 ± 3.38**

Table 1: **GO enrichment analysis (ES).** Performance of BraneMF compared to the baselines, measured by the ES. Standard deviation is computed for 20 runs of k-means clustering. Bold: best score, underlined: second best score. Parameters: $\gamma = 1, w = 10, d = 128$ for BraneMF, SM (Sec. 4) for baselines.

4.2. Clustering and GO enrichment of proteins

Proteins are building blocks of a biological system that facilitate cellular processes. Their discovery, functional annotation, and characterization are of great importance [45]. Therefore, we examine the ability of the learned features $\mathbf{\Omega}_d$ to cluster proteins of similar functions. Due to the large number of proteins (i.e., 4,900), we choose higher numbers for clusters, with $k \in \{40, 60, 80, 100\}$. Our choice of clustering algorithm is the k-means++ [1]. We execute it 20 times to take into account the randomness in the algorithm. For each of the obtained clusters, we perform Gene Set Enrichment Analysis (GSEA) [39] using the 'GO_Biological_Process_2018' and 'GO_Molecular_Function_2018' libraries of the *YeastEnrichr* database [23] consisting of 1,649 GO terms. A cluster is considered to be enriched if, at least, one GO term in a cluster is significantly enriched (adjusted P-value < 0.05). For all the significantly enriched clusters, the performance is measured by the enrichment score (ES) and Z-score [39]. The definition of these metrics is given in SM (Sec. 5). The final scores are calculated by computing the average over the 20 simulations. Similarly, we evaluate the clusters obtained for the baseline methods.

GSEA results with ES and Z-score are shown in Tables 1 and S6. The performance of BraneMF measured by ES is higher for $k = 60, 80$ and 100 compared to the baselines. For $k = 40$, deepNF achieves higher performance than BraneMF. Overall, this demonstrates that clusters

from BraneMF's features can group proteins into more significantly enriched biological processes for higher values of k. Moreover, Multi-node2vec and BraneExp are the second best-performing methods. The visual representation of the obtained clusters and their respective enrichment is given in SM (Fig. S1). This is our preliminary analysis to show the ability of the learned embeddings to cluster functionally related proteins. Nevertheless, the extensive analysis with different values of k could reveal the optimal k for each model. In the next section, we apply the learned embeddings to protein function prediction.

4.3. Protein function prediction

In this section, we investigate the reliability of learned features to predict protein functions. We model the problem of protein function prediction as a multi-label node classification task. We use the learned features, $\mathbf{\Omega}_d$, to train an SVM classifier and predict probability scores for each protein. We use the SVM implementation provided in the LIBSVMpackage [4]. To measure the performance of the SVM on the embedding vectors, we adopt a 5-fold cross validation (CV) process [7, 13]. We split all the annotated proteins into a training set, comprising of 80%, and a test set, comprising the remaining 20% ones. We train the SVM on the training set and predict the function of the test proteins. We use the standard radial basis kernel (RBF) for SVM and perform a nested 5-fold cross validation within the training set to select the optimal hyperparameters of the SVM via grid search (i.e., δ in the RBF kernel and the weight regularization parameter C). All performance results are averaged over 10 different CV trials. The evaluation metrics m-AUPR, M-AUPR, ACC, and F1 used for protein function prediction are mentioned in SM (Sec. 6.1).

We first investigate the added value of integration for protein function prediction. To do this, we have learned protein features for each input network and performed classification. We then compare the performance of the features learned from individual input networks to the integrated ones. The evaluation of results is done by computing the F1 score. The results for level I for BP, MF, and CC are shown in Fig. 2, while the results for levels II and III are shown in SM (Fig. S2). We observe that integration outperforms individual network protein function prediction. Also, the 'Experimental', 'Co-expression', and 'Database' networks demonstrate good performance in all

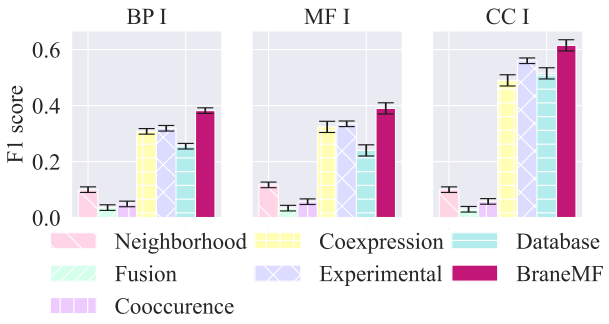


Figure 2: **Integrating multiple networks outperforms individual network.** Performance of BraneMF applied on individual yeast STRING networks, measured by the F1 score. Parameters: $\gamma = 1, w = 10, d = 128$. Error bars show the standard deviation across 10 CV trials.

three levels, whereas the 'Fusion' network gives the lowest score. This indicates the importance of the first three networks in the function prediction task, compared to the 'Neighborhood', 'Fusion', and 'Co-occurrence' networks.

Additionally, we have explored three different network integration strategies namely early, intermediate, and late. Early integration is performed before the modeling process, for example, merging all networks into one. On the contrary, late integration is done after the modeling process is applied to each network, and then it concatenates the obtained features. BraneMF is an intermediate integration model where integration is performed in the learning process of embedding computation. To show the effectiveness of the intermediate level of integration, we have compared BraneMF with BraneMF-early and BraneMF-late. In BraneMF-early, the PPMI matrix is computed from the adjacency matrix of the network obtained by taking the union of all six network layers. Then, d-dimensional protein features are learned. In BraneMF-late, the protein features are learned independently for each layer, and the final features are obtained by taking their average. The performance is evaluated by computing the F1 score and accuracy metrics. As we can observe in Fig. 3 and SM (Fig. S3), BraneMF outperforms the rest integration strategies for all three levels of BP, MF, and CC. There is an increase of 2% in the accuracy of BP I when compared to BraneMF-early and an increase of 10% compared to the BraneMF-late integration model.

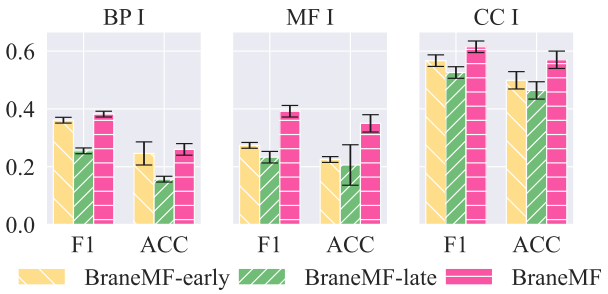


Figure 3: **Integration strategies.** Performance of BraneMF compared to early and late integration measured by F1 score. Parameters: $\gamma = 1, w = 10, d = 128$. The error bars show the standard deviation across 10 CV trials.

Method	BP I	MF I	CC I
SNF	0.199 ± 0.01	0.104 ± 0.00	0.206 ± 0.01
Mashup	0.271 ± 0.0	0.263 ± 0.02	0.520 ± 0.02
deepNF	0.341 ± 0.01	0.342 ± 0.02	0.564 ± 0.02
Multi-Net	0.335 ± 0.01	0.353 ± 0.02	0.532 ± 0.02
Multi-n2v	0.331 ± 0.01	0.323 ± 0.01	0.511 ± 0.01
OhmNet	0.321 ± 0.01	0.300 ± 0.01	0.512 ± 0.01
MultiVERSE	0.312 ± 0.01	0.294 ± 0.01	0.502 ± 0.02
BraneExp	<u>0.352 ± 0.01</u>	<u>0.368 ± 0.01</u>	0.548 ± 0.03
Graph2GO	0.340 ± 0.01	0.355 ± 0.01	<u>0.564 ± 0.02</u>
BraneMF	**0.382 ± 0.01**	**0.392 ± 0.02**	**0.615 ± 0.02**

Table 2: **Protein function prediction.** Performance of BraneMF, compared to baseline methods using the F1 score. Parameters: $\gamma = 1, w = 10, d = 128$ for BraneMF, SM (Sec. 4) for baselines. The standard deviation is computed based on 10 CV trials. Bold: best score, underlined: second best score.

Also, the performance of BraneMF for MF and CC is significantly higher than BraneMF-early and BraneMF-late under F1 and ACC scoring schemes. Hence, BraneMF 's improvement can be partially attributed to the fact that separately computing the random walk matrices of each individual layer uncovers compressed topological patterns, that are difficult to identify in the combined network (BraneMF-early model) where different edge types are not distinguished. Moreover, BraneMF has the advantage over late integration to benefit from capturing inter-layer correlation of modalities at the feature level that is challenging for late integration.

We also compare the performance of BraneMF to the baseline methods with the validation strategies described earlier. Table 2 and SM (Tables S7, S8, and S9) show the classification results for level I, II, and III of BP, MF, and CC respectively. We observe that protein function prediction based on BraneMF substantially outperforms other integration methods in assigning a previously unseen protein to its known functional categories in a CV experiment. For instance, the F1 score for BraneMF (BP I) is 38.2%, that is 3 points higher than BraneExp and 4.2 points higher than Graph2GO, the second best-performing method. Whereas BraneMF correctly assigned 26% of proteins (on average) to BP I category, in contrast to 24.9% for Graph2GO and 22% for BraneExp. Similarly, BraneMF consistently outperforms the baselines for level II and level III.

4.4. Protein-Protein interaction (PPI) prediction

The interactome is the complete map of PPIs that can occur in an organism. It is still an open question whether a complete interactome of any organism will ever be discovered by experimental techniques [21]. Therefore, predictive methods have become more popular in systems biology to reveal the wiring patterns of proteins. Effective integration of PPIs from different data sources (experimental and/or computational) can help us to have a near complete set of interactome [21]. In this task, our goal is to predict the missing (unseen) PPIs (edges) between proteins (nodes) using the learned features. We use PPIs from the 2015 and 2021 STRING networks to form training and test sets, respectively. We form the positive training set from PPIs that did not change from 2015 to 2021, and the positive test set from the PPIs that did not exist in 2015 but gained existence in 2021. The same number of PPIs that do not exist in both networks are sampled to generate negative instances for each training and test sets respectively. The learned embeddings (Sec. 3) of protein u and v, given as $\Omega_d[u]$ and $\Omega_d[v]$, are converted into edge feature vectors by applying the coordinate-wise Hadamard product or cosine similarity operations [15]. Definitions of these operations are given in SM (Sec. 7.1). We perform the prediction task using logistic regression classifier with L2 regularization. The performance of PPI prediction is evaluated based on AUROC (area under the Receiver Operating Characteristic curve) and AUPR (area under the

Method	AUPR-H	AUROC-H	AUPR-C	AUROC-C
SNF	0.637	0.628	0.575	0.559
Mashup	0.757	0.743	<u>0.712</u>	**0.707**
deepNF	0.764	0.747	0.490	0.480
Multi-Net	0.735	0.724	0.490	0.480
Multi-n2v	0.526	0.528	0.511	0.509
OhmNet	0.513	0.514	0.516	0.516
MultiVERSE	0.500	0.501	0.501	0.501
BraneExp	<u>0.777</u>	**0.760**	0.683	0.680
Graph2GO	0.721	<u>0.757</u>	0.502	0.498
BraneMF	**0.783**	0.747	**0.725**	<u>0.682</u>

Table 3: **PPI prediction performance.** Performance of BraneMF, compared to the baseline methods, measured by the AUROC and AUPR for the edge features computed by coordinate-wise operations given by Hadamard product (-H) and cosine similarity (-C). Bold: best score, underlined: second best score. Parameters: $\gamma = 1, w = 10, d = 128$ for BraneMF, SM (Sec. 4) for baselines.

Precision-Recall curve). The results are shown in Table 3. We observe that BraneMF has competitive and consistent behavior across almost all evaluation metrics for the PPI prediction, achieving 1.5% higher performance (AUPR-H) than BraneExp which is the second best performing model. deepNF and Mashup also perform well under specific evaluation metrics.

Additionally, we reconstructed the yeast STRING network using the learned representations. The details are provided in SM (Sec. 7.2). The respective results are shown in SM (Fig. S4). Here, we observe that BraneMF outperforms all baseline models for both evaluation metrics. For the top $1,000$ edges, notably all the baseline methods except SNF, give 100% of Precision. When we increase the number of edges to 1 million, BraneMF and Multi-node2vec continue to show higher performance when compared to the baseline models.

5. Discussion

The wide availability of omics data has driven the need for the development of novel integrative methods that can properly analyse and interpret them. We have presented BraneMF, an integrative framework for analyzing the topology of multiple protein-protein interaction networks towards extracting relevant protein features from heterogeneous data sources. BraneMF performs integration of multilayer biological networks with the concept of joint matrix factorization, that generalizes random walk-based network embedding models. More precisely, BraneMF brings the best of two worlds: expressiveness of the well-celebrated random walk-based embedding models (e.g., DeepWalk [33], node2vec [15]) and the solid formulation of matrix factorization—going further by extending them to integrate multiple sources. We have demonstrated the wide applicability of BraneMF in exploiting functional analysis of proteins in PPI networks by studying the quality of clusteredness of functionally related proteins, the accuracy of predicting protein functions, and the inference of interactions in the reconstruction of the yeast interactome. Besides, while comparing against nine other baseline models, BraneMF has shown competitive performance in all downstream assessments. In a modeling framework that integrates multiple sources, it is imperative to define the uncertainty of the model's predictions.

We have performed sensitivity analysis for three parameters, namely γ, w, and d that BraneMF depends on. The embedding size range is consistent with the current literature. The selected sizes of the embedding vectors are $\{128, 256, 512, 1024\}$ for BraneMF and all the baseline models. Also, w and γ are given as $\{2, 4, 6, 8, 10\}$ and $\{0, 0.25, 0.50, 0.75, 1\}$, respectively. The details are presented in SM (Sec. 8). We observed that BraneMF performs relatively consistently, even with smaller dimension sizes i.e., 128. For other networks with a smaller or larger number of nodes, the embedding dimensions used are mostly selected empirically. The trade-off is between accuracy and computational time. Large embedding sizes may potentially increase the performance in the downstream tasks, since the vectors could capture extended aspects of a node. Yet, higher dimensions drastically affect computing time and parametrization effort. Therefore, for smaller networks we believe $d = 128$ could be an ideal choice, while for larger networks, such as the PPI networks for human (with approximately $25,000$ genes), the size can be increased from 128 to 512 or 1024 depending on the task and computing capacity.

In summary, we conclude that BraneMF is simpler, depends on less parameters, and produces results comparable, if not better, to more complex methods (e.g., deepNF). Although our formulation is expressive enough to capture these representations, its multiscale properties have certain limitations. First, the model learns one global representation that coalesces all possible scales of net-

work relationships. Hence, different scales of the representation are not independently accessible. Additionally, our approach lacks to capture long-range node dependencies (i.e., higher values of w) which could be interesting to study [5]. As future work, we intend to conflate additional protein associations such as post-transcriptional and post-translation regulation information that may impact the functional relationships of proteins in the real world. Besides, it is also possible to upgrade BraneMF to take into account protein (node) features such as biochemical properties and protein sequences in the learning process [51]. These data types can provide insights towards more accurate predictions for functional analysis of proteins. The functionality and applicability of BraneMF are beyond embedding proteins thus not limited to biological networks. BraneMF is a versatile tool that provides an effective, unified, and scalable network integration framework with diverse applications.

Funding. This work has been supported in part by ANR (French National Research Agency) under the JCJC project GraphIA (ANR-20-CE23-0009-01).

References

[1] Arthur, D., Vassilvitskii, S., 2007. k-means++: the advantages of careful seeding, in: Proc. Annu. ACM-SIAM Symp. Discrete Algorithms, p. 1027–1035.

[2] Bagavathi, A., Krishnan, S., 2018. Multi-Net: a scalable multiplex network embedding framework, in: Aiello, L., Cherifi, C., Cherifi, H., Lambiotte, R., Lió, P., Rocha, L. (Eds.), Proc. Int. Conf. Complex Netw. Appl, pp. 119–131.

[3] Çelikkanat, A., Malliaros, F.D., 2020. Exponential family graph embeddings, in: Proc. AAAI Conf. Artif. Intell., pp. 3357–3364.

[4] Chang, C.C., Lin, C.J., 2011. LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. 2, 1–27.

[5] Chanpuriya, S., Musco, C., 2020. InfiniteWalk: Deep network embeddings as Laplacian embeddings with a nonlinearity, in: Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 1325–1333.

[6] Chicco, D., Jurman, G., 2020. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. BMC Genomics 21, 1–13.

[7] Cho, H., Berger, B., Peng, J., 2016. Compact integration of multi-network topology for functional analysis of genes. Cell Syst. 3, 540–548.

[8] Consortium, G.O., 2004. The Gene Ontology (GO) database and informatics resource. Nucleic acids research 32, D258–D261.

[9] Cozzetto, D., Buchan, D.W.A., Bryson, K., Jones, D.T., 2013. Protein function prediction by massive integration of evolutionary analyses and multiple data sources, in: BMC Bioinformatics, p. S1.

[10] Di Nanni, N., Bersanelli, M., Milanesi, L., Mosca, E., 2020. Network diffusion promotes the integrative analysis of multiple omics. Front. Genet. 11.

[11] Dong, X., Frossard, P., Vandergheynst, P., Nefedov, N., 2012. Clustering with multi-layer graphs: A spectral perspective. IEEE Trans. Signal Process. 60, 5820–5831.

[12] Fan, K., Guan, Y., Zhang, Y., 2020. Graph2GO: a multi-modal attributed network embedding method for inferring protein functions. GigaScience 9.

[13] Gligorijević, V., Barot, M., Bonneau, R., 2018. deepNF: deep network fusion for protein function prediction. Bioinformatics 34, 3873–3881.

[14] Gligorijević, V., Pržulj, N., 2015. Methods for biological data integration: perspectives and challenges. J. R. Soc. Interface 12, 20150571.

[15] Grover, A., Leskovec, J., 2016. node2vec: Scalable feature learning for networks, in: Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 855–864.

[16] Guo, Q., Cozzo, E., Zheng, Z., Moreno, Y., 2016. Lévy random walks on multiplex networks. Sci. Rep. 6.

[17] Hamilton, W.L., Ying, R., Leskovec, J., 2017. Representation learning on graphs: Methods and applications. IEEE Data Eng. Bull. 40, 52–74.

[18] Hu, X., Hu, Y., Wu, F., Leung, R.W.T., Qin, J., 2020. Integration of single-cell multi-omics for gene regulatory network inference. Comput. Struct. Biotechnol. J. 18, 1925–1938.

[19] Irizarry, R.A., Wang, C., Zhou, Y., Speed, T.P., 2009. Gene set enrichment analysis made simple. Statistical methods in medical research 18, 565–575.

[20] Jagtap, S., Çelikkanat, A., Pirayre, A., Bidard, F., Duval, L., Malliaros, F.D., 2021. Multiomics data integration for gene regulatory network inference with exponential family embeddings, in: Proc. Eur. Sig. Image Proc. Conf., pp. 1221–1225.

[21] Keskin, O., Tuncbag, N., Gursoy, A., 2016. Predicting protein-protein interactions from the molecular to the proteome level. Chem. Rev. 116, 4884–4909.

[22] Kipf, T.N., Welling, M., 2016. Variational graph auto-encoders, in: Bayesian Deep Learning Workshop (NIPS 2016).

[23] Kuleshov, M.V., Jones, M.R., Rouillard, A.D., Fernandez, N.F., Duan, Q., Wang, Z., Koplev, S., Jenkins, S.L., Jagodnik, K.M., Lachmann, A., McDermott, M.G., Monteiro, C.D., Gundersen, G.W., Ma'ayan, A., 2016. Enrichr: a comprehensive gene set enrichment analysis web server 2016 update. Nucleic Acids Res. 44, W90–W97.

[24] Levy, O., Goldberg, Y., 2014. Neural word embedding as implicit matrix factorization. Adv. Neural Inf. Process. Syst. 27, 2177–2185.

[25] Li, Y., Wu, F.X., Ngom, A., 2018. A review on machine learning principles for multi-view biological data integration. Brief. Bioinform. 19, 325–340.

[26] Liu, W., Chen, P.Y., Yeung, S., Suzumura, T., Chen, L., 2017. Principled multilayer network embedding, in: Proc. IEEE Int. Conf. Data Min. Workshops (ICDMW), pp. 134–141.

[27] Liu, Z., Huang, C., Yu, Y., Fan, B., Dong, J., 2020. Fast attributed multiplex heterogeneous network embedding, in: Proc. ACM Int. Conf. Inf. Knowl. Manag., pp. 995–1004.

[28] Luo, Y., Zhao, X., Zhou, J., Yang, J., Zhang, Y., Kuang, W., Peng, J., Chen, L., Zeng, J., 2017. A network integration approach for drug-target interaction prediction and computational drug repositioning from heterogeneous information. Nature communications 8, 1–13.

[29] Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J., 2013. Distributed representations of words and phrases and their compositionality, in: Proc. Ann. Conf. Neur. Inform. Proc. Syst., pp. 3111–3119.

[30] Nguyen, D., Malliaros, F.D., 2018. BiasedWalk: Biased sampling for representation learning on graphs, in: IEEE Int. Conf. Big Data, pp. 4045–4053.

[31] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Vanderplas, J., 2011. scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12.

[32] Peng, J., Xue, H., Wei, Z., Tuncali, I., Hao, J., Shang, X., 2021. Integrating multi-network topology for gene function prediction using deep neural networks. Brief. Bioinform. 22, 2096–2105.

[33] Perozzi, B., Al-Rfou, R., Skiena, S., 2014. DeepWalk: Online learning of social representations, in: Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 701–710.

[34] Pio-Lopez, L., Valdeolivas, A., Tichit, L., Remy, É., Baudot, A., 2021. MultiVERSE: a multiplex and multiplex-heterogeneous network embedding approach. Sci. Rep. 11, 8794.

[35] Qiu, J., Dong, Y., Ma, H., Li, J., Wang, K., Tang, J., 2018. Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec, in: Proc. ACM Int. Conf. Web Search Data Min., pp. 459–467.

[36] Rudolph, M., Ruiz, F., Mandt, S., Blei, D., 2016. Exponential family embeddings, in: Proc. Ann. Conf. Neur. Inform. Proc. Syst., pp. 478–486.

[37] Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T., 2003. Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome research 13, 2498–2504.

[38] Solé-Ribalta, A., De Domenico, M., Gómez, S., Arenas, A., 2016. Random walk centrality in interconnected multilayer networks. Physica D 323, 73–79.

[39] Subramanian, A., Tamayo, P., Mootha, V.K., Mukherjee, S., Ebert, B.L., Gillette, M.A., Paulovich, A., Pomeroy, S.L., Golub, T.R., Lander, E.S., et al., 2005. Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc. Nat. Acad. Sci. U.S.A. 102, 15545–15550.

[40] Subramanian, I., Verma, S., Kumar, S., Jere, A., Anamika, K., 2020. Multi-omics data integration, interpretation, and its application. Bioinform. Biol. Insights 14, 1–24.

[41] Szklarczyk, D., Gable, A.L., Nastou, K.C., Lyon, D., Kirsch, R., Pyysalo, S., Doncheva, N.T., Legeay, M., Fang, T., Bork, P., Jensen, L.J., von Mering, C., 2021. The STRING database in 2021: customizable protein–protein networks, and functional characterization of user-uploaded gene/measurement sets. Nucleic Acids Res. 49, D605–D612.

[42] Tsitsulin, A., Mottin, D., Karras, P., Müller, E., 2018. VERSE: Versatile graph embeddings from similarity measures, in: Proc. World Wide Web Conf., pp. 539–548.

[43] Wang, B., Mezlini, A.M., Demir, F., Fiume, M., Tu, Z., Brudno, M., Haibe-Kains, B., Goldenberg, A., 2014. Similarity network fusion for aggregating data types on a genomic scale. Nat. Meth. 11, 333–340.

[44] Wilson, J.D., Baybay, M., Sankar, R., Stillman, P., Popa, A.M., 2021. Analysis of population functional connectivity data via multilayer network embeddings. Network Science 9, 99–122.

[45] Wood, V., Lock, A., Harris, M.A., Rutherford, K., Bähler, J., Oliver, S.G., 2019. Hidden in plain sight: what remains to be discovered in the eukaryotic proteome? Open biology 9, 180241.

[46] Yan, J., Risacher, S.L., Shen, L., Saykin, A.J., 2018. Network approaches to systems biology analysis of complex disease: integrative methods for multi-omics data. Brief. Bioinform. 19, 1370–1381.

[47] Yue, X., Wang, Z., Huang, J., Parthasarathy, S., Moosavinasab, S., Huang, Y., Lin, S.M., Zhang, W., Zhang, P., Sun, H., 2020. Graph embedding on biomedical networks: methods, applications and evaluations. Bioinformatics 36, 1241–1251.

[48] Zhang, H., Qiu, L., Yi, L., Song, Y., 2018. Scalable multiplex network embedding., in: Proc. IJCAI Int. Jt. Conf. Artif. Intell., pp. 3082–3088.

[49] Zhang, Z., Zhang, J., 2009. A big world inside small-world networks. PloS One 4, e5686.

[50] Zhou, G., Li, S., Xia, J., 2020a. Network-based approaches for multi-omics integration. Methods in molecular biology (Clifton, N.J.) 2104, 469–487.

[51] Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020b. Graph neural networks: A review of methods and applications. AI Open 1, 57–81.

[52] Zitnik, M., Leskovec, J., 2017. Predicting multicellular function through multi-layer tissue networks. Bioinformatics 33, i190–i198.

Supplementary Material

1. Pseudocode of BraneMF

Algorithm 1 BraneMF

Input: Multilayer graph $\mathcal{G} = \{\mathcal{G}_l\}_{l=1}^{L}$;
Parameters: window size: w, embedding dimension: d, and weighting factor: γ
Output: d-dimension protein features, $\mathbf{\Omega}_d$

1. For each $\mathcal{G}_l$, obtain its degree matrix $\mathbf{D}^{(l)}$ and adjacency matrix $\mathbf{A}^{(l)}$
2. Compute power matrix $\mathbf{P}^{(l)}, (\mathbf{P}^{(l)})^2, \ldots, (\mathbf{P}^{(l)})^w$ for each l in $\mathcal{G}_l$ (where $\mathbf{P}^{(l)} = (\mathbf{D}^{(l)})^{-1}\mathbf{A}^{(l)}$)
3. Compute PPMI matrix $\mathbf{M}^{(l)}$ for $\mathcal{G}$ as given in Eq. (1)
4. Solve the optimization problem in Eq. (2) to obtain $\mathbf{U}$ and $\bar{\mathbf{\Sigma}}$
5. Compute protein features $\mathbf{\Omega}_d = \mathbf{U}_d(\bar{\mathbf{\Sigma}}_d)^{\gamma}$

return $\mathbf{\Omega}_d$

2. Functional derivative of O

To obtain $\mathbf{U}$ and $\mathbf{V}$, we solve the objective function O in Eq. (2) using the derivation given below:

$$\frac{\partial O}{\partial \mathbf{U}} = -\left(\sum_{l=1}^{L}(\mathbf{M}^{(l)} - \mathbf{U}\mathbf{\Sigma}^{(l)}\mathbf{V}^{\top})\right)\mathbf{V}\mathbf{\Sigma}^{(l)} + \alpha\mathbf{U} + \beta(\mathbf{U}\mathbf{V}^{\top} - \mathbf{I})\mathbf{V}^{\top},$$

$$\frac{\partial O}{\partial \mathbf{V}^{\top}} = \mathbf{\Sigma}^{(l)}\mathbf{U}^{\top}\left(\sum_{i=1}^{L}(\mathbf{M}^{(l)} - \mathbf{U}\mathbf{\Sigma}^{(l)}\mathbf{V}^{\top})\right) + \alpha\mathbf{V}^{\top} + \beta\mathbf{U}^{\top}(\mathbf{U}\mathbf{V}^{\top} - \mathbf{I}). \tag{S1}$$

3. Datasets

3.1. Yeast PPI networks

Network	Nodes	Edges	Density	Evidence
Neighborhood	4,900	7,656	8.741×10^{-3}	Gene order and sequence homology
Fusion	4,900	492	3.943×10^{-3}	Orthology and fusion
Cooccurrence	4,900	1,231	3.861×10^{-3}	Orthology
Coexpression	4,900	54,317	6.562×10^{-3}	Gene expression data
Experimental	4,900	48,190	5.600×10^{-3}	Biochemical, biophysical genetic experiments
Database	4,900	29,231	5.946×10^{-3}	Human curation

Table S1: **Overview of the datasets used in our study**. We select the network edges whose combined score exceeds 900.

3.2. Functional annotations

The functional annotations were downloaded from the Gene Ontology database [8] (May 2022 update). Each category of GO is represented in levels (i.e., level I, II, and III). A lower level (i.e., Level I) represents more specific terms whereas a higher one (i.e., Level III) represents more general terms. Table S2 shows the number of terms per category.

Terms	Level I	Level II	Level III
Biological Process (BP)	855	535	244
Molecular Function (MF)	216	126	53
Cellular Component (CC)	181	113	54

Table S2: **Overview of the Gene Ontology (GO) terms used for prediction**. Level I: 10 < proteins per term < 30; Level II: 30 < proteins per term < 100; and Level III: 100 < proteins per term < 300.

4. Baseline models

We have selected nine multi-layer network integration methods to benchmark BraneMF. A brief introduction and the details of their implementation are given below. We test all the methods including BraneMF with the datasets of a well-studied organism, i.e., yeast (tax id: 4392). Our implementation takes as input the taxonomy id of each organism and fetches the recently available data from the STRING database [41]. Moreover, we perform extensive parameter tuning for each baseline method. The details of parameter choices for each method are given in Sec. 4.2 of the supplementary material. We report the best performance for each model in the main results, while the remaining results are provided in the following sections of this supplementary material.

4.1. Description and implementation

1. SNF [43]: Similarity network fusion (SNF) is a computational method for data integration. SNF first constructs a similarity network of samples (e.g., patients) for each available data type and then efficiently fuses them into one network. We have applied SNF to integrate our datasets. Since we already have PPI networks, we have skipped the network construction step. We obtain the integrated (fused) network from SNF with the best parameters given by the authors. Moreover, we need features for each node of this integrated network. To obtain features, we have performed SVD on the graph and obtain node features as the first d columns of the left singular matrix U, where d is the embedding size.

   Implementation: http://compbio.cs.toronto.edu/SNF/SNF/Software_files/SNFtool_v2.1.tar.gz
   http://compbio.cs.toronto.edu/SNF/SNF/Software_files/SNFmatlab_v2.1.zip

2. Mashup [7]: Mashup extracts a compact vector representation of the network that could explain topological patterns of nodes in multiple heterogeneous interaction networks.

   Mashup is closely related to BraneMF except for some perceptible differences:

   - Firstly, node proximities in Mashup are captured using a random walk with restart (RWR) matrix. In BraneMF, we have chosen to utilize PPMI matrices that are computed from standard random walk (RW) matrices without restart probabilities. Specifically, in Mashup, the RWR for a node $i \in \mathcal{V}$ is defined by

   $$\mathbf{s}_i^{t+1} = (1 - p_r)\mathbf{P}\mathbf{s}_t^t + p_r\mathbf{e}_i, \tag{S2}$$

   where p_r is the return probability and t is the number of steps. The matrix $\mathbf{S} := [\mathbf{s}_1^\infty | \dots | \mathbf{s}_i^\infty | \dots | \mathbf{s}_N^\infty] \in \mathbb{R}^{N \times N}$ is constructed, where each $\mathbf{s}_i^\infty$ is the column vector for node $i \in \mathcal{V}$. Each entry of $\mathbf{s}_i^\infty$ indicates the probability of visiting j by starting from node i over an infinite walk with an additional return probability. For BraneMF, the choice of the PPMI matrix is inspired by the relationship between SkipGram-based random walk embeddings (e.g., DeepWalk) and matrix factorization [35]. This formulation has shown significant improvement in single-layer graph embedding methods [35]. Taking into account this inspiration, we have utilized the concept of jointly factorizing PPMI matrices for multilayer graph embedding. The PPMI matrix is defined in Eq. (1) of the main paper. Besides, the PPMI formulation has a window size parameter (T), which is used to control the similarity definition between nodes. Thus, it introduces a flexible framework which can be adapted depending on the structural properties of a given network.
   - Secondly, Mashup and BraneMF rely on different computational optimization techniques to obtain integrative embeddings. Mashup has two instances for learning embeddings. (i) A multinomial logistic model for dimension reduction by integrating the diffusion states of each node in each network layer. (ii) Optimization based on singular value decomposition (SVD). The later instance of Mashup is close to BraneMF's joint

matrix factorization. Mashup simply concatenates the normalized RWR matrices, and then it performs SVD [7]. More precisely, the resulting concatenated matrix $\mathbf{S}$ has a dimension of $NL \times L$. The logarithm of $\mathbf{S}$ is given as $\tilde{\mathbf{S}}$ and truncated SVD on $\tilde{\mathbf{S}}$ is performed (with a user-specified number of components) to get a low-rank factorization $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$. On the contrary, we propose an efficient way to combine the spectra of PPMI matrices, whose result can be viewed as the joint spectrum shared by all the graph layers. For a multilayer graph $\mathcal{G}$ composed of L layers, the PPMI matrices $\mathbf{M}^{(l)}$ are computed for each $l \in \{1, \ldots, L\}$. This set of $\mathbf{M}^{(l)}$ matrices are jointly decomposed to singular vector $\mathbf{\Sigma}^{(l)}$ and singular value matrices ($\mathbf{U}$ and $\mathbf{V}$) that are shared across all layers (see Sec. 3.2 in the main paper).

Implementation: https://groups.csail.mit.edu/cb/mashup/mashup.tar.gz

3. deepNF [13]: deepNF is a network fusion method based on a multimodal deep autoencoder (MDA) to integrate different heterogeneous networks into a compact, low-dimensional feature representation common to all networks.

   Implementation: https://github.com/VGligorijevic/deepNF

4. MultiNet [2]: MultiNet is a fast scalable multiplex network embedding framework. First, it computes random walks on all nodes across multiple network layers. Then it merges all sequences of random walks in one document and learns node features by using an optimization that maximizes the likelihood of neighbors of a node across network layers.

5. Multi-node2vec [44]: Multi-node2vec is a fast network embedding model for multilayer/multiplex networks that identifies a continuous and low-dimensional representation for the unique nodes in the network. It is an extension of the Node2Vec model for multilayer networks.

   Implementation: https://github.com/jdwilson4/multi-node2vec

6. OhmNet [52]: OhmNet is a hierarchy-aware unsupervised learning approach for multi-layer networks. It learns node features from a multi-layer network, where each layer represents a protein-protein interaction network and all these networks are in a tree structure. Since our dataset has no hierarchy, we have used a flat tree where the network hierarchy is flattened; PPI networks are not rendered inside of each other, but instead are rendered as sibling hierarchy elements (i.e. sharing the same parent node) toward features in the common parent in the hierarchy.

   Implementation: https://github.com/mims-harvard/ohmnet

7. MultiVERSE [34]: MultiVERSE is an extension of the VERSE framework using Random Walks with Restart on Multiplex (RWR-M) and Multiplex-Heterogeneous (RWR-MH) networks. Since all our networks share the same set of nodes, we have used the RWR-M model of MultiVERSE for integrative analysis.

   Implementation: https://github.com/LPioL/MultiVERSE

8. Graph2GO [12]: A graph-based representation learning method for predicting protein functions. Graph2GO uses the VGAE [22] model to learn embeddings from each layer and perform concatenation to obtain final embeddings.

   Implementation : https://github.com/yanzhanglab/Graph2GO

9. BraneExp [20]: It is a network integration framework that considers expressive conditional probability models to relate nodes within random walk sequences. The model uses exponential family distributions to capture interactions between nodes in random walks that traverse nodes within and across input network layers, learning node features using exponential family graph embeddings.

   Implementation : https://github.com/Surabhivj/BRANE-EXP

4.2. Parameter selection

All multi-layer network integration methods that are based on machine learning and mathematical models require tuning a certain set of parameters to learn protein features. From the model description given by each method in their respective research article, we highlight parameters that could be tuned to improve the model performance. The remaining parameters that have little impact on performance have been set to the default values. In Table S3, we provide the required parameters for each method that is tuned. Note that the representation in Table S3 is simplified to show the dependency of baseline methods on the different parameters. For some methods, a direct comparison of parameters is not possible since they may share different parameter spaces. We adopt this approach to simplify the parameter selection strategy.

Method	d	w	l	n	σ	e	γ	r	k	b
SNF	✓	✓	✗	✗	✗	✗	✗	✗	✓	✗
Mashup	✓	✗	✗	✗	✗	✗	✗	✓	✗	✗
DeepNF	✓	✗	✗	✗	✗	✓	✗	✓	✓	✓
MultiNet	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
Multi-n2v	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
OhmNet	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
MultiVERSE	✓	✓	✗	✗	✓	✗	✗	✓	✓	✗
Graph2GO	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗
BraneExp	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
BraneMF	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗

Table S3: **Overview of parameters considered for tuning**. Green coloured ticks indicate that the method depends on the respective parameters. Red crosses show that method does not depend on a particular parameter.

1. Embedding size (d): size of protein feature vector. Its dimensionality is typically much lower than that of the ambient space. We chose $d \in \{128, 256, 512, 1024\}$.
2. Walk length (l): it is a parameter to select the length of a node set you would like to obtain while performing random walks on a graph. For instance, a *walk* of length 5 is defined as "proteinA proteinB proteinC proteinD proteinX". We chose $l \in \{15, 20\}$.
3. Window size (w): the number of nodes (proteins) that will be used to determine the context of each node (protein). For instance, in a *walk* of length 3, such as, "proteinA proteinB proteinC", a window size of 2 would mean your samples are like (proteinA proteinB) or (proteinB proteinC). We chose $w \in \{2, 4, 6, 8, 10\}$.
4. Number of walks (n): this parameter allows to select the number of random walks that will be sampled per node (protein). We chose $n \in \{10, 20\}$.

5. Learning rate (σ): it is a hyper-parameter that controls how much we are adjusting the weights in the learning process with respect to the gradient of the loss function. A lower σ represents a smaller step along the downward slope. We chose $\sigma \in \{0.1, 0.01, 0.001, 0.0001\}$.
6. Number of epochs (e): an epoch is one learning cycle where the learner sees the whole training data set and calculates the error rate. We chose $e \in \{60, 80\}$.
7. Restart probability (r): this parameter is used in models that rely on Random Walks with Restart (RWR). While performing random walks, at each iteration, the walker can also restart by jumping to any randomly selected node in the graph, with a defined restart probability. We chose $r \in \{0.8, 0.85, 0.9, 0.95\}$.
8. Gamma (γ): it is a weighting factor used by our BraneMF model. It represents the power to be applied on the singular values ($\bar{\mathbf{\Sigma}}$) used for the computation of the embeddings (please see line 5 of Alg. 1). We chose $\gamma \in \{0, 0.25, 0.50, 0.75, 1\}$.
9. Number of samples (k): it is the parameter to choose the number of times we would like to perform Random Walk with Restart (RWR). We chose $k \in \{2, 3, 4, 6\}$.
10. Batch size (b): it is the number of samples that will be used for training at one time. We chose $b \in \{32, 64, 128\}$.

Method	**Parameters**
SNF	$k = 6, w = 10$
Mashup	$r = 0.8$
deepNF	$b = 64; k = 4; r = 0.95; e = 80$
MultiNet	$l = 20; n = 20; w = 10$
Multi-n2v	$l = 20; n = 20; w = 10$
OhmNet	$w = 10; l = 20; n = 10$
MultiVERSE	$w = 10; k = 4; r = 0.95, \sigma = 0.01$
Graph2GO	$e = 80; \sigma = 0.01$
BraneExp	$l = 20; n = 10; w = 10$
BraneMF	$w = 10; \gamma = 1$

Table S4: **Parameter selection I.** The table shows the best performing parameters for the clustering and protein function prediction tasks.

Method	**Parameters**
SNF	$k = 6, w = 10$
Mashup	$r = 0.95$
deepNF	$b = 64; k = 4; r = 0.95; e = 80$
MultiNet	$l = 20; n = 20; w = 10$
Multi-n2v	$l = 10; n = 20; w = 2$
OhmNet	$w = 2; l = 15; n = 10$
MultiVERSE	$w = 2; k = 2; r = 0.8, \sigma = 0.01$
Graph2GO	$e = 80; \sigma = 0.01$
BraneExp	$l = 15; n = 20; w = 10$
BraneMF	$w = 2; \gamma = 0.5$

Table S5: **Parameter selection II.** The table shows the best performing parameters for the PPI prediction and network reconstruction tasks.

5. Gene Ontology (GO) enrichment for clusters

5.1. Enrichment Score (ES)

Consider a gene set G_k, where $k = 1, \ldots, K$. G_k consists of a list of n_k genes (g_{kj}), i.e., $G_k = \{g_{kj} : j = 1, \ldots, n_k\}$. Each gene in the set is represented in the ranked list L. The set of genes outside of G_k is defined as as $\bar{G}_k = \{\bar{g}_{kj} : 1, \ldots, n - n_k\}$. The Enrichment Score (ES) for a given gene set G_k is given as:

$$ES = \sup_{1 \le i \le n} \left(F_i^{G_k} - F_i^{\bar{G}_k} \right),$$

where sup$(\cdot)$ is the supremum, i represents the position in list L, and

$$F_i^{G_k} = \frac{\sum_{t=1}^{i} |s_t|^\alpha . \mathbb{1}_{gene_t \in G_k}}{\sum_{t=1}^{n} |s_t|^\alpha . \mathbb{1}_{gene_t \in G_k}},$$

$$F_i^{\bar{G}_k} = \frac{\sum_{t=1}^{i} |s_t|^\alpha . \mathbb{1}_{gene_t \in \bar{G}_k}}{n - n_k},$$

where $\mathbb{1}$ is the indicator function for the membership in a given gene set. s_t is given by the correlation of g_{kj} and weighted by α [39].

5.2. Z-Score

The Z-score for gene set G_k is given by:

$$\text{Z-score} = \sqrt{n_k}\bar{t}, \text{with } \bar{t} = \frac{1}{n_k} \sum_{i \in G_k} t_i,$$

where n_k is number of genes in G_k and t_i is the t-statistic that is referred to signal to noise ratio for each gene.

The detailed description of the scoring metrics are given in [39, 19]

5.3. Results

Method	$k = 40$	$k = 60$	$k = 80$	$k = 100$
		Z-score		
SNF	-1.73 ± 0.07	-1.80 ± 0.03	-1.77 ± 0.01	-1.81 ± 0.02
Mashup	-1.85 ± 0.14	-1.88 ± 0.12	-1.84 ± 0.03	-1.92 ± 0.02
deepNF	-2.10 ± 0.07	-2.09 ± 0.06	-2.03 ± 0.01	-2.02 ± 0.03
MultiNet	-1.87 ± 0.03	-1.99 ± 0.02	-1.97 ± 0.02	-2.05 ± 0.02
Multi-n2v	-2.62 ± 0.22	-2.05 ± 0.06	-1.92 ± 0.04	-1.96 ± 0.02
OhmNet	-1.96 ± 0.03	-1.92 ± 0.02	-2.03 ± 0.01	-1.99 ± 0.01
MultiVERSE	-1.93 ± 0.04	-1.97 ± 0.02	-2.00 ± 0.03	-2.06 ± 0.02
BraneExp	-1.98 ± 0.04	-1.98 ± 0.04	-1.99 ± 0.02	-2.00 ± 0.02
Graph2GO	-2.06 ± 0.04	-1.89 ± 0.04	-2.02 ± 0.03	-2.04 ± 0.02
BraneMF	-1.92 ± 0.04	-1.93 ± 0.03	-1.98 ± 0.03	-2.01 ± 0.03

Table S6: **GO enrichment analysis of clusters.** Performance of BraneMF compared to the baselines, measured by Z-scores. Standard deviation is computed for all 20 runs of k-means clustering. Parameters: $\gamma = 1, w = 10, d = 128$ for BraneMF, Sec. 4 for baselines.

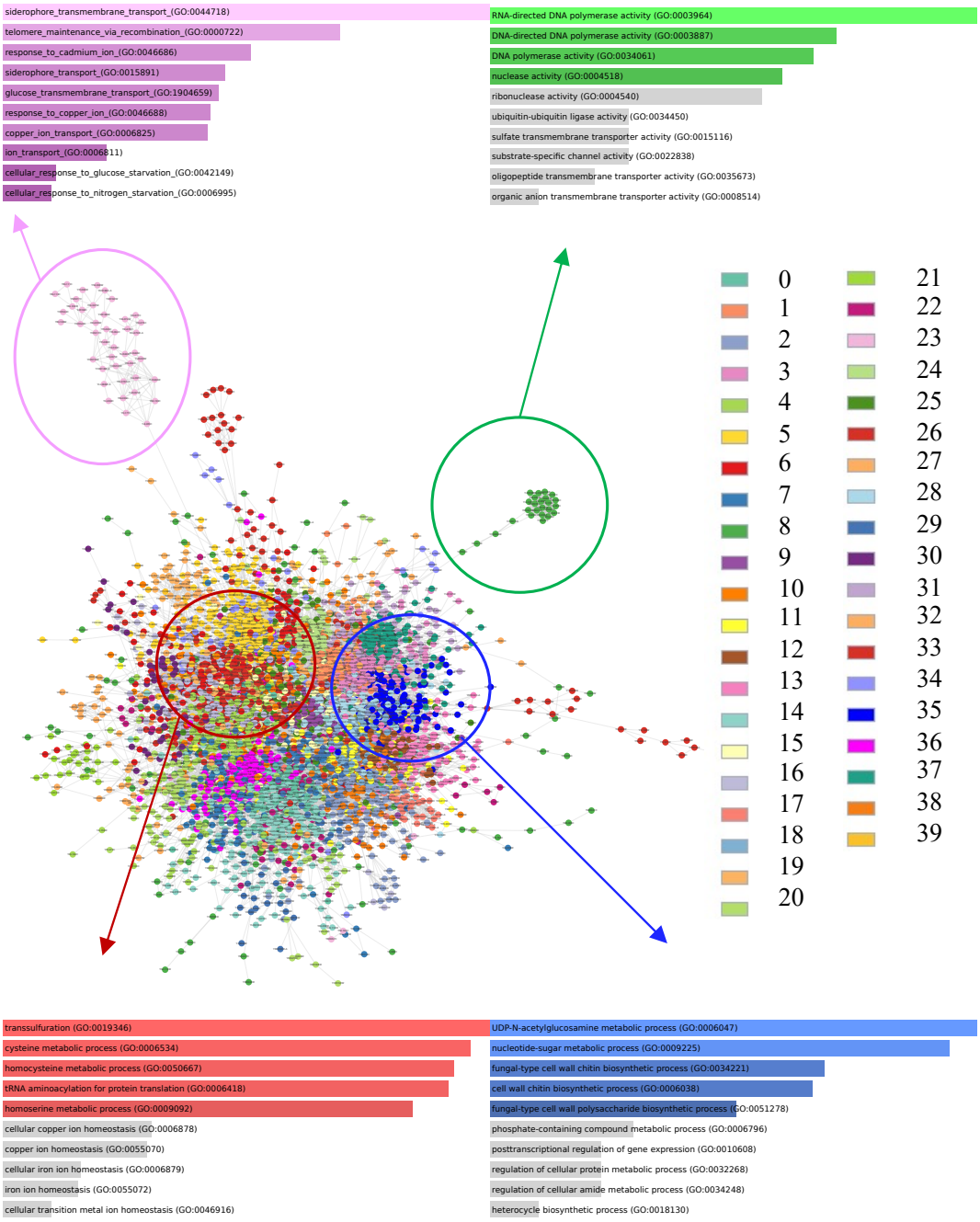


Figure S1: **Visual representation of clusters and their GO enrichment.** We present an example of clustering results of the first simulation of the k-means++ algorithm for $k = 40$. The obtained clusters are then mapped to the integrated STRING network (combined score > 900) visualised using Cytoscape [37] with the default layout. Node colours represent the 40 different clusters. To show the GO enrichment results, we have selected four clusters and their respective enrichment is shown by yeast 'enrichr' barplot results (sorted by p-value). [23]. Note that, to compare with the baselines, we perform k-means++ 20 times and compute the ES and Z-scores of significantly enriched clusters for each simulation. Finally, we report the mean ES and Z-score with their standard deviation across 20 simulations (Table 1).

6. Protein function prediction

6.1. Evaluation metrics

We use the following metrics to evaluate the prediction performance:

- The micro-averaged area under the Precision-Recall curve (m-AUPR) is calculated by vectorizing the matrices of the protein and predicted scores of its function and known binary annotations, then computing the AUPR by using these two vectors.
- The Macro-AUPR (M-AUPR) is computed using the AUPR for each function separately, and then averaging these values across all functions.
- Accuracy (ACC) measures the percentage of test proteins that were correctly predicted.
- Micro-averaged F1 score (F1) is computed by assigning the top three predictions to each protein with micro average parameter, and computing the F1 score [13].

6.2. Added value of integration in comparison to individual networks

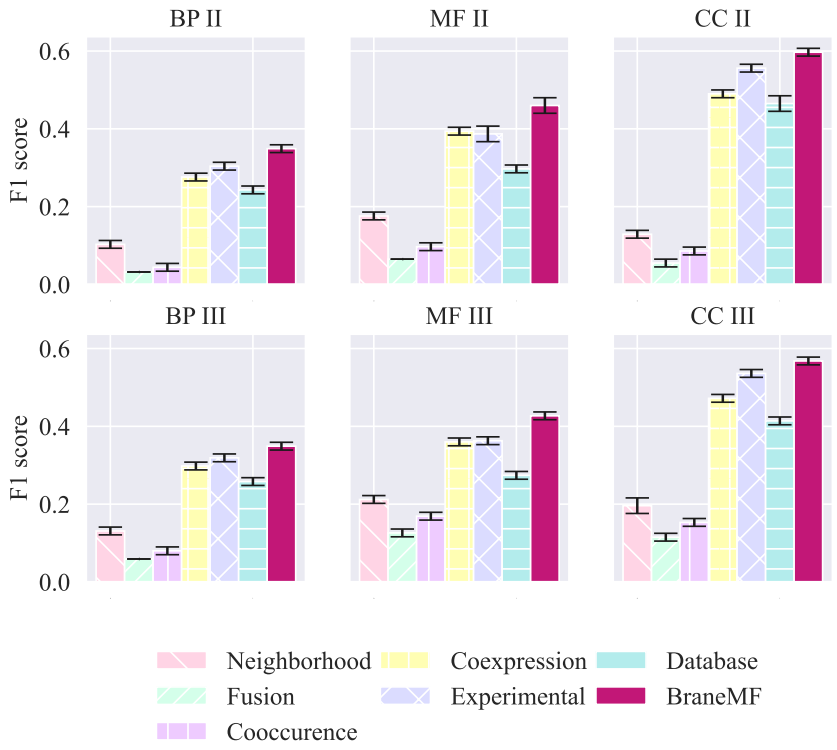


Figure S2: **Integrating multiple networks outperforms individual networks in protein function prediction.** We compare the cross-validation performance of BraneMF on individual yeast STRING networks, measured by F1 score. Parameters: $\gamma = 1, w = 10, d = 128$. The error bars show the standard deviation across 10 CV trials.

6.3. Classification performance for early, intermediate, and late integration

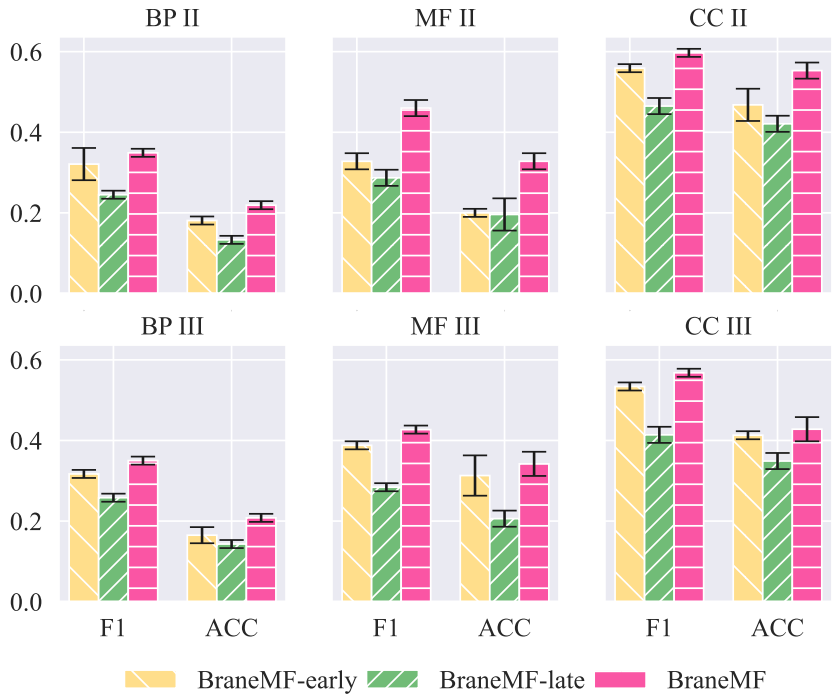


Figure S3: **Integration strategies.** Performance of BraneMF compared to early and late integration, measured by the F1 score. Parameters: $\gamma = 1, w = 10, d = 128$. The error bars show the standard deviation across 10 CV trials.

6.4. Protein function prediction compared to baseline methods

Method	m-AUPR	M-AUPR	F1	ACC
		BP I		
SNF	0.176 ± 0.01	0.224 ± 0.01	0.199 ± 0.01	0.150 ± 0.00
Mashup	0.362 ± 0.02	0.240 ± 0.01	0.277 ± 0.00	0.161 ± 0.01
deepNF	0.427 ± 0.02	0.260 ± 0.01	0.341 ± 0.01	0.211 ± 0.02
MultiNet	0.415 ± 0.02	0.257 ± 0.01	0.335 ± 0.01	0.212 ± 0.03
Multi-n2v	0.417 ± 0.02	0.250 ± 0.01	0.331 ± 0.01	0.201 ± 0.02
OhmNet	0.361 ± 0.02	0.255 ± 0.01	0.321 ± 0.01	0.063 ± 0.01
MultiVERSE	0.353 ± 0.02	0.223 ± 0.02	0.312 ± 0.01	0.117 ± 0.01
BraneExp	0.454 ± 0.02	0.280 ± 0.01	0.352 ± 0.01	0.220 ± 0.08
Graph2GO	0.458 ± 0.02	0.279 ± 0.01	0.340 ± 0.01	0.249 ± 0.02
BraneMF	**0.504 ± 0.02**	**0.303 ± 0.01**	**0.382 ± 0.01**	**0.260 ± 0.02**
		BP II		
SNF	0.220 ± 0.01	0.260 ± 0.01	0.220 ± 0.01	0.140 ± 0.01
Mashup	0.385 ± 0.02	0.337 ± 0.01	0.260 ± 0.01	0.130 ± 0.01
deepNF	0.464 ± 0.01	0.381 ± 0.01	0.309 ± 0.01	0.154 ± 0.01
MultiNet	0.458 ± 0.02	0.378 ± 0.01	0.323 ± 0.01	0.178 ± 0.01
Multi-n2v	0.494 ± 0.01	0.406 ± 0.01	0.329 ± 0.01	0.171 ± 0.01
OhmNet	0.382 ± 0.01	0.325 ± 0.01	0.285 ± 0.01	0.027 ± 0.01
MultiVERSE	0.387 ± 0.02	0.329 ± 0.01	0.293 ± 0.01	0.093 ± 0.01
BraneExp	0.474 ± 0.02	0.391 ± 0.01	0.322 ± 0.01	0.204 ± 0.03
Graph2GO	0.487 ± 0.02	0.398 ± 0.02	0.317 ± 0.01	0.185 ± 0.03
BraneMF	**0.524 ± 0.02**	**0.424 ± 0.02**	**0.349 ± 0.01**	**0.219 ± 0.02**
		BP III		
SNF	0.167 ± 0.00	0.224 ± 0.01	0.153 ± 0.01	0.052 ± 0.01
Mashup	0.484 ± 0.02	0.450 ± 0.01	0.289 ± 0.01	0.144 ± 0.01
deepNF	0.535 ± 0.01	0.478 ± 0.01	0.318 ± 0.01	0.157 ± 0.01
MultiNet	0.555 ± 0.01	0.496 ± 0.01	0.343 ± 0.01	0.189 ± 0.01
Multi-n2v	0.560 ± 0.02	0.504 ± 0.01	0.341 ± 0.01	0.185 ± 0.02
OhmNet	0.439 ± 0.01	0.411 ± 0.01	0.300 ± 0.01	0.010 ± 0.00
MultiVERSE	0.455 ± 0.02	0.422 ± 0.01	0.315 ± 0.01	0.079 ± 0.01
BraneExp	0.537 ± 0.01	0.495 ± 0.01	0.330 ± 0.01	0.183 ± 0.02
Graph2GO	0.568 ± 0.01	0.509 ± 0.01	0.329 ± 0.01	0.162 ± 0.01
BraneMF	**0.585 ± 0.01**	**0.526 ± 0.01**	**0.350 ± 0.01**	**0.208 ± 0.01**

Table S7: **Protein function prediction (BP)**. Performance of BraneMF, compared to baseline methods for BP using m-AUPR, M-AUPR, F1 and ACC scores. Parameters: $\gamma = 1, w = 10, d = 128$ for BraneMF, SM (Sec. 4) for baselines. The standard deviation is computed based on 10 CV trials.

Method	m-AUPR	M-AUPR	F1	ACC
		MF I		
SNF	0.192 ± 0.01	0.120 ± 0.01	0.104 ± 0.00	0.142 ± 0.01
Mashup	0.255 ± 0.02	0.192 ± 0.01	0.263 ± 0.02	0.183 ± 0.04
deepNF	0.388 ± 0.03	0.235 ± 0.02	0.342 ± 0.02	0.273 ± 0.02
MultiNet	0.376 ± 0.02	0.249 ± 0.02	0.353 ± 0.02	0.306 ± 0.01
Multi-n2v	0.398 ± 0.03	0.203 ± 0.01	0.323 ± 0.01	0.262 ± 0.02
OhmNet	0.293 ± 0.02	0.211 ± 0.01	0.300 ± 0.01	0.020 ± 0.02
MultiVERSE	0.294 ± 0.03	0.192 ± 0.01	0.294 ± 0.01	0.145 ± 0.01
BraneExp	0.410 ± 0.02	0.256 ± 0.01	0.368 ± 0.01	0.303 ± 0.11
Graph2GO	0.404 ± 0.02	0.243 ± 0.02	0.355 ± 0.01	0.287 ± 0.11
BraneMF	**0.457 ± 0.04**	**0.278 ± 0.02**	**0.392 ± 0.02**	**0.350 ± 0.03**
		MF II		
SNF	0.185 ± 0.01	0.214 ± 0.01	0.126 ± 0.01	0.123 ± 0.00
Mashup	0.362 ± 0.02	0.310 ± 0.01	0.345 ± 0.02	0.228 ± 0.01
deepNF	0.428 ± 0.02	0.335 ± 0.01	0.396 ± 0.01	0.233 ± 0.01
MultiNet	0.440 ± 0.03	0.350 ± 0.02	0.416 ± 0.02	0.267 ± 0.04
Multi-n2v	0.447 ± 0.05	0.350 ± 0.03	0.398 ± 0.03	0.224 ± 0.06
OhmNet	0.342 ± 0.02	0.285 ± 0.01	0.334 ± 0.01	0.038 ± 0.01
MultiVERSE	0.363 ± 0.03	0.303 ± 0.02	0.348 ± 0.02	0.116 ± 0.01
BraneExp	0.463 ± 0.03	0.368 ± 0.02	0.436 ± 0.02	0.294 ± 0.10
Graph2GO	0.455 ± 0.02	0.359 ± 0.01	0.420 ± 0.01	0.292 ± 0.04
BraneMF	**0.518 ± 0.02**	**0.404 ± 0.02**	**0.460 ± 0.02**	**0.328 ± 0.02**
		MF III		
SNF	0.155 ± 0.01	0.147 ± 0.01	0.165 ± 0.01	0.037 ± 0.00
Mashup	0.393 ± 0.02	0.347 ± 0.02	0.333 ± 0.01	0.221 ± 0.02
deepNF	0.442 ± 0.03	0.389 ± 0.02	0.367 ± 0.01	0.236 ± 0.02
MultiNet	0.481 ± 0.03	0.419 ± 0.02	0.397 ± 0.02	0.309 ± 0.01
Multi-n2v	0.457 ± 0.03	0.406 ± 0.02	0.333 ± 0.02	0.150 ± 0.05
OhmNet	0.365 ± 0.02	0.343 ± 0.02	0.323 ± 0.01	0.014 ± 0.00
MultiVERSE	0.364 ± 0.02	0.337 ± 0.02	0.329 ± 0.01	0.131 ± 0.01
BraneExp	0.517 ± 0.03	0.454 ± 0.02	0.417 ± 0.02	**0.345 ± 0.02**
Graph2GO	0.501 ± 0.02	0.448 ± 0.02	0.396 ± 0.01	0.338 ± 0.02
BraneMF	**0.541 ± 0.03**	**0.473 ± 0.02**	**0.427 ± 0.01**	0.342 ± 0.02

Table S8: **Protein function prediction (MF)**. Performance of BraneMF, compared to baseline methods for MF using m-AUPR, M-AUPR, F1 and ACC scores. Parameters: $\gamma = 1, w = 10, d = 128$ for BraneMF, SM (Sec. 4) for baselines. The standard deviation is computed based on 10 CV trials.

Method	m-AUPR	M-AUPR	F1	ACC
		CC I		
SNF	0.178 ± 0.03	0.234 ± 0.02	0.206 ± 0.01	0.048 ± 0.01
Mashup	0.681 ± 0.02	0.414 ± 0.02	0.520 ± 0.02	0.432 ± 0.04
deepNF	0.715 ± 0.02	0.423 ± 0.02	0.564 ± 0.02	0.461 ± 0.02
MultiNet	0.662 ± 0.03	0.394 ± 0.02	0.532 ± 0.02	0.431 ± 0.06
Multi-n2v	0.663 ± 0.02	0.378 ± 0.03	0.511 ± 0.01	0.411 ± 0.02
OhmNet	0.590 ± 0.02	0.380 ± 0.02	0.512 ± 0.01	0.020 ± 0.03
MultiVERSE	0.586 ± 0.04	0.365 ± 0.02	0.502 ± 0.02	0.249 ± 0.08
BraneExp	0.694 ± 0.05	0.418 ± 0.03	0.548 ± 0.03	0.472 ± 0.03
Graph2GO	0.732 ± 0.03	0.438 ± 0.03	0.564 ± 0.02	0.490 ± 0.03
BraneMF	**0.812 ± 0.02**	**0.470 ± 0.03**	**0.615 ± 0.02**	**0.570 ± 0.03**
		CC II		
SNF	0.258 ± 0.02	0.323 ± 0.02	0.258 ± 0.02	0.040 ± 0.00
Mashup	0.681 ± 0.08	0.604 ± 0.02	0.505 ± 0.03	0.414 ± 0.03
deepNF	0.733 ± 0.01	0.617 ± 0.01	0.550 ± 0.01	0.431 ± 0.01
MultiNet	0.724 ± 0.02	0.610 ± 0.02	0.555 ± 0.01	0.453 ± 0.02
Multi-n2v	0.723 ± 0.01	0.592 ± 0.01	0.523 ± 0.01	0.458 ± 0.02
OhmNet	0.640 ± 0.02	0.544 ± 0.02	0.513 ± 0.01	0.093 ± 0.03
MultiVERSE	0.628 ± 0.02	0.529 ± 0.02	0.504 ± 0.01	0.249 ± 0.01
BraneExp	0.749 ± 0.02	0.630 ± 0.02	0.559 ± 0.01	0.462 ± 0.04
Graph2GO	0.749 ± 0.01	0.631 ± 0.01	0.549 ± 0.01	0.481 ± 0.04
BraneMF	**0.806 ± 0.02**	**0.666 ± 0.02**	**0.597 ± 0.01**	**0.553 ± 0.02**
		CC III		
SNF	0.364 ± 0.02	0.374 ± 0.02	0.342 ± 0.01	0.041 ± 0.01
Mashup	0.620 ± 0.01	0.555 ± 0.01	0.471 ± 0.01	0.345 ± 0.01
deepNF	0.655 ± 0.01	0.564 ± 0.01	0.508 ± 0.01	0.357 ± 0.02
MultiNet	0.688 ± 0.02	0.603 ± 0.03	0.546 ± 0.01	0.400 ± 0.02
Multi-n2v	0.660 ± 0.02	0.595 ± 0.02	0.491 ± 0.01	0.286 ± 0.05
OhmNet	0.588 ± 0.02	0.523 ± 0.03	0.493 ± 0.02	0.091 ± 0.01
MultiVERSE	0.598 ± 0.01	0.535 ± 0.02	0.496 ± 0.01	0.224 ± 0.01
BraneExp	0.706 ± 0.01	0.634 ± 0.01	0.559 ± 0.01	0.378 ± 0.02
Graph2GO	0.701 ± 0.02	0.623 ± 0.02	0.544 ± 0.01	0.387 ± 0.02
BraneMF	**0.734 ± 0.01**	**0.646 ± 0.02**	**0.568 ± 0.01**	**0.428 ± 0.03**

Table S9: **Protein function prediction (CC)**. Performance of BraneMF, compared to baseline methods for CC using m-AUPR, M-AUPR, F1 and ACC scores. Parameters: $\gamma = 1, w = 10, d = 128$ for BraneMF, SM (Sec. 4) for baselines. The standard deviation is computed based on 10 CV trials.

7. Protein-protein interaction (PPI) prediction

7.1. Computation of edge features

Consider nodes u and v. To compute features for a candidate edge between node u and v, we first extract node features $\Omega[u]$ and $\Omega[v]$ respectively. Then, we perform coordinate wise operations (Hadamard product, cosine distance), as follows:

1. Hadamard product:

$$\Omega[u]_i \times \Omega[v]_i$$

2. Cosine distance:

$$1 - \frac{\Omega[u]_i \cdot \Omega[v]_i}{||\Omega[u]_i||||\Omega[v]_i||}$$

$\Omega[u]_i$ is the i^{th} index the embedding vector $\Omega[u]$ of node u.

7.2. Network reconstruction

To reconstruct the yeast STRING network using the learned representations, we first compute pairwise node similarities using the normalized inner product (also known as cosine similarity) of the trained embeddings. Then, a candidate PPI network is constructed, whose nodes are proteins and the edges represent the similarity between two proteins, weighted by the similarity score. The total number of possible edges of this reconstructed network is 20.4 million. We evaluate this reconstructed network using yeast STRING PPI of year 2021 (reference network). This network has 1 million edges. For a fair comparison, we narrow down our evaluation to the 1 million top-scoring edges of the reference network. The performance is measured using the precision@k. This metric gives us the number of true edges (edges in the reference network) that are present in the top-k edges of the reconstructed network. We choose to compute precision@k for each non-negative power of 10 for all the models [31]. Recent studies have suggested that when the dataset is unbalanced (the number PPIs in a PPI network is much smaller than the number of non-PPIs), accuracy could provide an overoptimistic estimation of the classifier's ability on the majority class (non-PPIs) [6]. Therefore, to have a more comprehensive evaluation, we also measure Matthews correlation coefficient (MCC) for several thresholds. The results for both metrics are shown in Fig. S4. It is observed that BraneMF outperforms all baseline models for both evaluation metrics. Biological networks show small-world properties [49]. These properties can be extracted by focusing on important edges in the graph. For the top $1,000$ edges, notably all the baseline methods except SNF, give 100% of Precision. When we increase the number of edges to 1 million, BraneMF continues to show high performance when compared to the baseline models. This demonstrates that BraneMF can capture information accurately even for big networks.

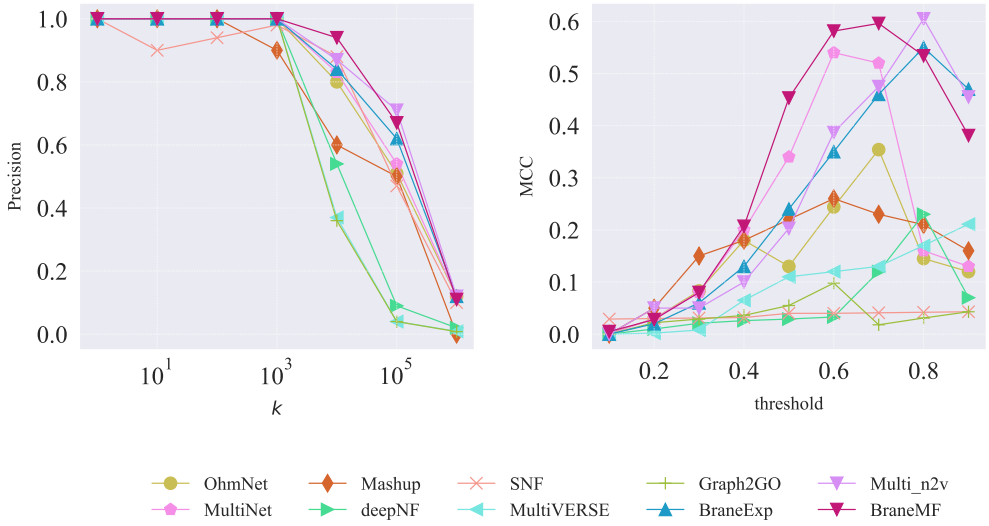


Figure S4: **Network reconstruction.** Performance of BraneMF for the network reconstruction task, compared against baseline models. Left plot: the *x*-axis represents the number of top *k* PPIs and the *y*-axis shows the precision (edges present in the reference network) for these PPIs. Right plot: the *x*-axis represents the threshold for selecting the edges and the *y*-axis shows the MCC for the respective threshold.

8. Uncertainty analysis

To assess the goodness of our results, we have performed various uncertainty analysis tests. Since protein function prediction is the most critical task, we chose to perform parameter sensitivity analysis for it. Our protein features depend on the setting of the γ, w, and d parameters. Therefore, we set the weighting factor γ as $[0, 0.25, 0.50, 0.75, 1]$, the window size w as $[2, 4, 6, 8, 10]$, and the dimension d as $[128, 256, 512, 1024]$. For each of the selected values of γ and d, we learn protein features and perform classification using 5-fold cross-validation across 10 trials. The performance (F1 score and accuracy) w.r.t. w when $\gamma = 1$ and $d = 128$ is shown in Figure S5. We observe that the F1 score for $w = 6, 8$, and 10 is higher than the one for $w = 2$ and 4. However, the accuracy for $w = 2$ is higher for BP and MF. Additionally, $w = 6$ and 10 show higher performance in the accuracy of CC compared to other values of window sizes. In our main results, we keep $w = 10$ considering all tasks, however for new datasets we recommend setting it to $6, 8$ or 10 depending on the downstream task. The results for different values of γ for level I, II, and III of BP, MF, and CC are shown in Figure S6. We have observed that the performance when $\gamma = 1$ is higher for all the runs compared to other values of γ. Besides, 0.5 is the second best performing value of γ. For new datasets, we recommend to test the performance for $\gamma = 0.5$ and 1.

Lastly, the embedding dimension is one of the essential parameters considered to test the performance of the model and compare against the baselines. We compare the performance of BraneMF with all baseline models for $d = \{128, 256, 512, 1024\}$. We have used a common heuristic approach to pick the range of the embedding size as a power of 2. Powers of 2 will increase cache utilization during data movement, thus reducing bottlenecks. For other networks with fewer or more nodes, the choice of the embedding dimension is mostly done empirically. The trade-off is between accuracy and computational concerns. An embedding vector of higher dimensionality may increase the accuracy in a downstream task, since the vectors can capture more aspects of the nodes. But more dimensions could require a higher computing time and memory resources. The results for the classification performance (F1 score) with respect to d for level I, II, and III of BP, MF, and CC are shown in Figure S7. We observe that the performance of BraneMF is quite stable compared to other methods for all values of d. BraneMF performs well even at lower dimensions. The performance of Mashup, SNF, deepNF, Graph2GO, and MultiVERSE increases with d, while BraneMF, BraneExp, MultiNet, OhmNet, and Multi-node2vec perform better at lower dimensions than higher ones.

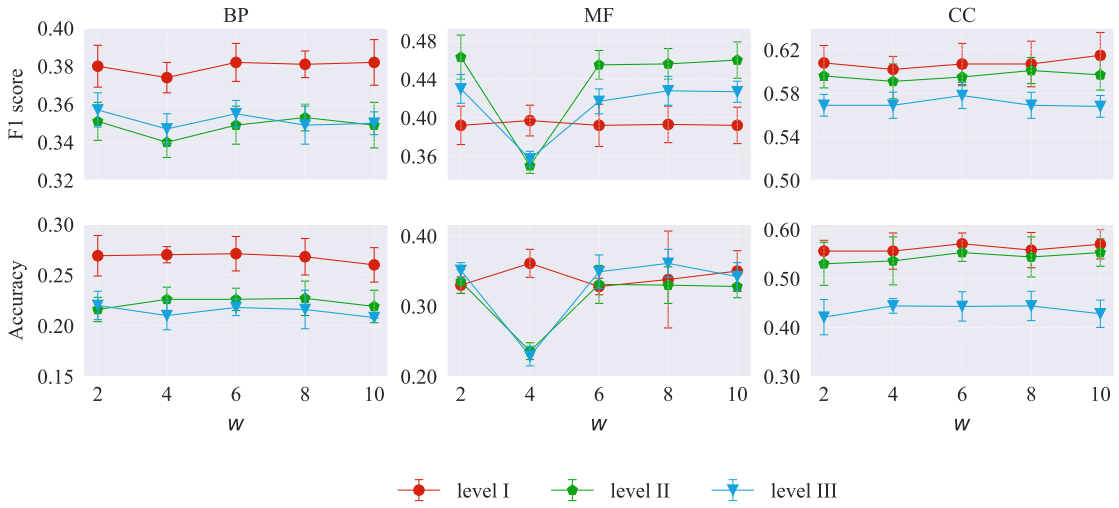


Figure S5: **Cross-validation performance.** Performance (F1 score and accuracy) of BraneMF with $w \in [2, 4, 6, 8, 10]$, $\gamma = 1$ and $d = 128$, in 5-fold cross validation for function prediction, compared to baseline network integration methods. Error bars represent standard deviations across 10 CV trials.

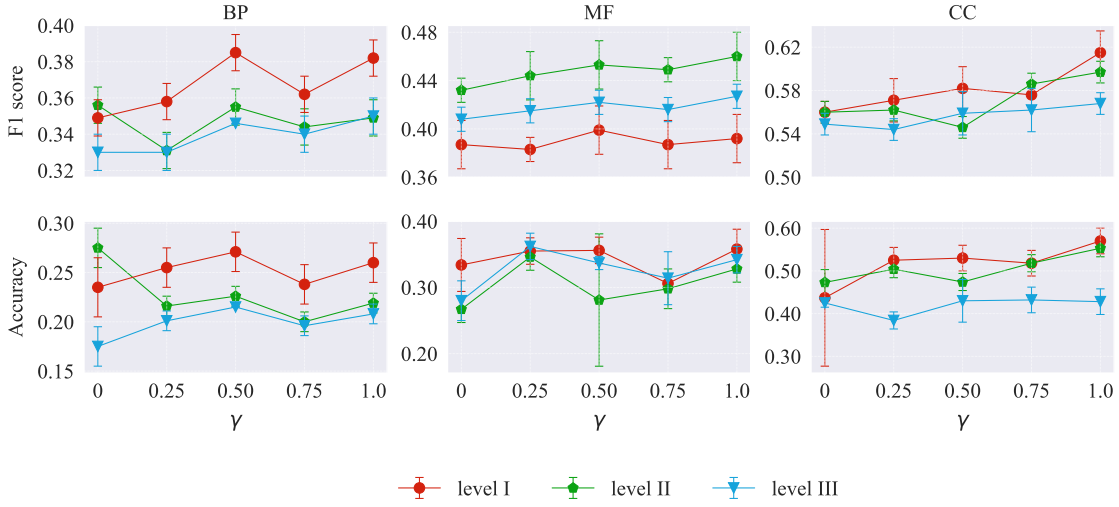
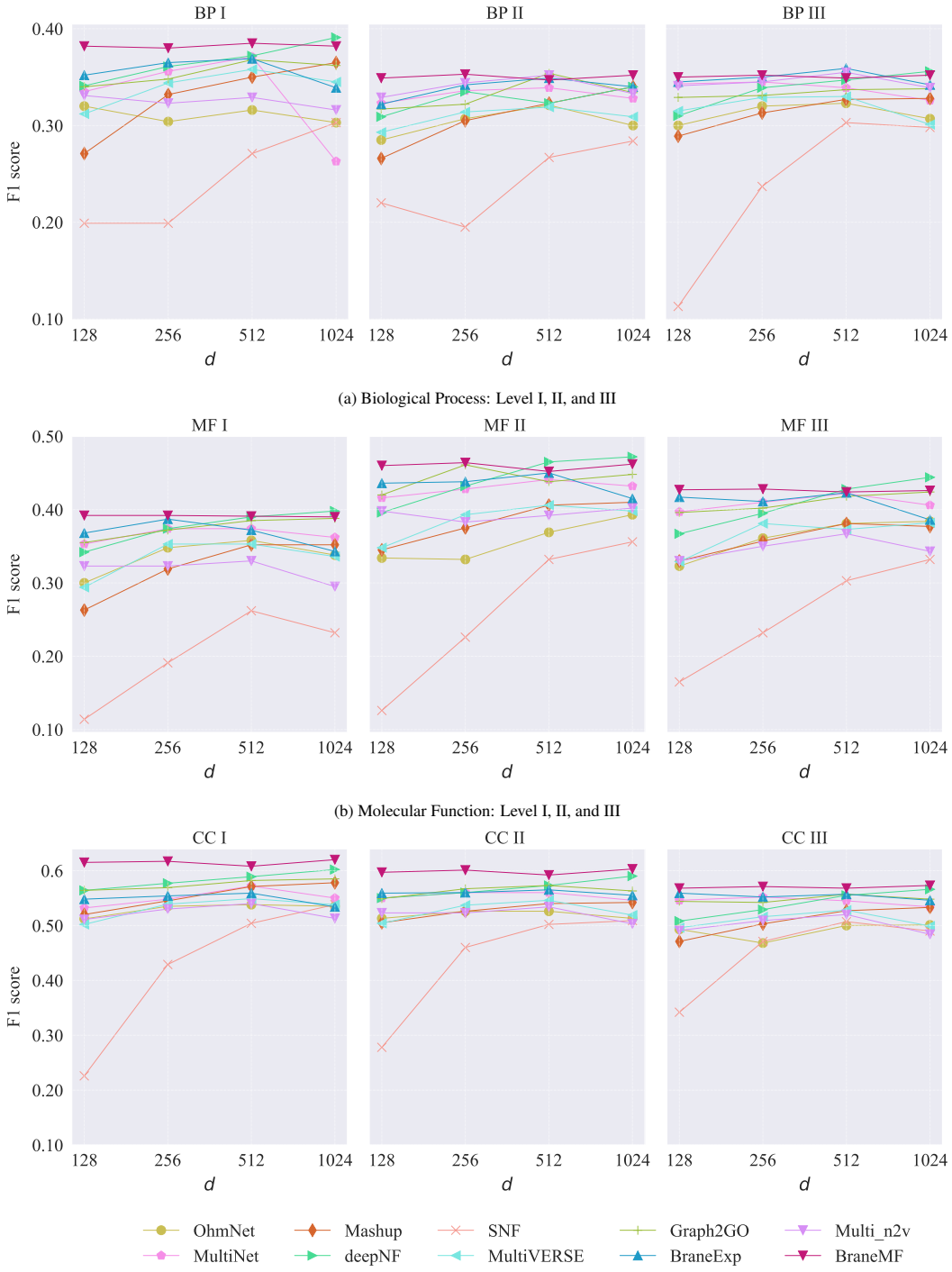


Figure S6: **Cross-validation performance.** Performance (F1 score and accuracy) of BraneMF with $\gamma \in [0, 0.25, 0.50, 0.75, 1.0]$, $w = 10$ and $d = 128$, in 5-fold cross validation for function prediction, compared to baseline network integration methods. Error bars represent standard deviations across 10 CV trials.



(a) Biological Process: Level I, II, and III

(b) Molecular Function: Level I, II, and III

(c) Cellular Component: Level I, II, and III

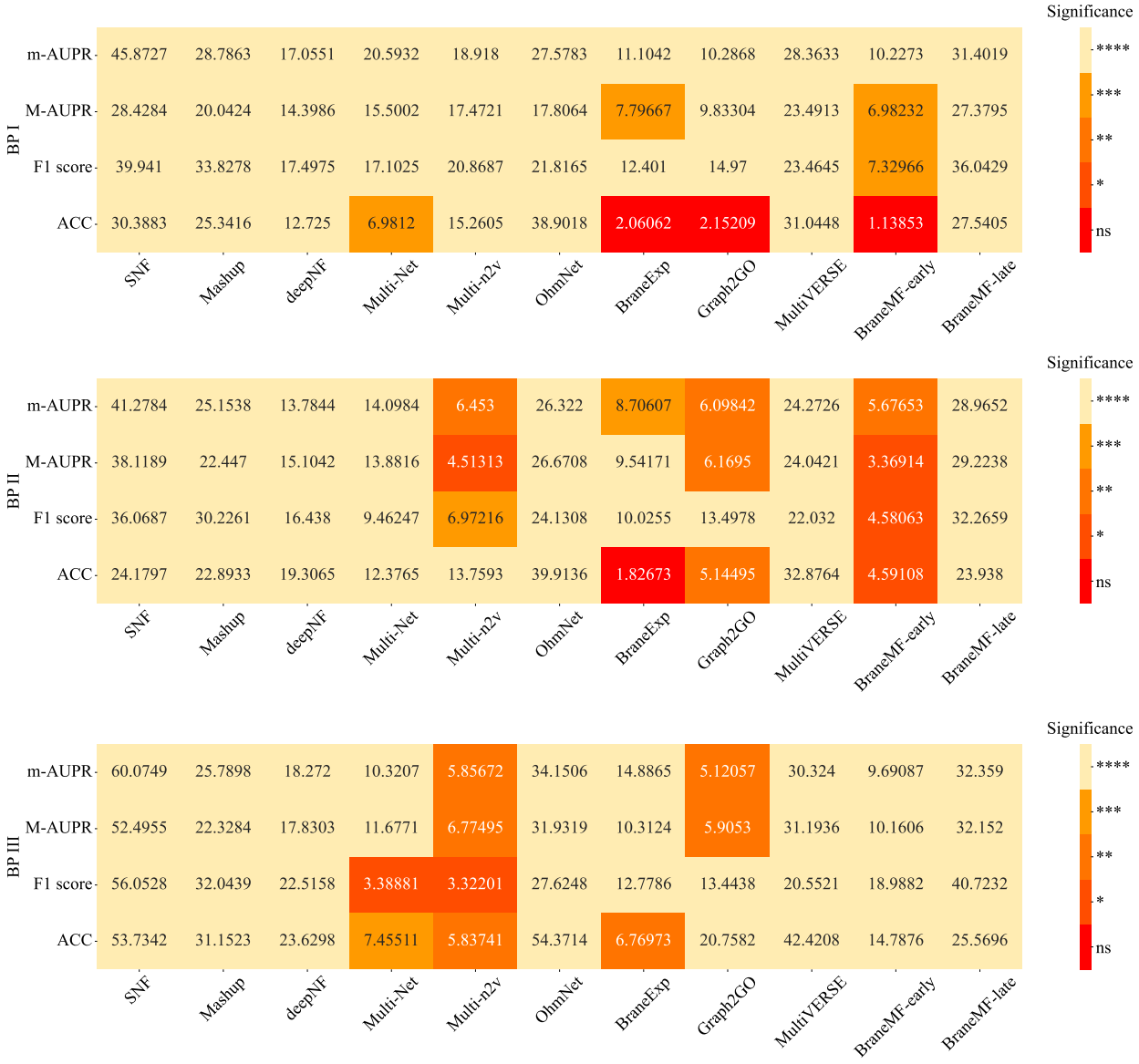
Figure S7: **Classification performance benchmark (BraneMF versus baselines)**. x-axis: protein feature dimension d; y-axis: F1 score.

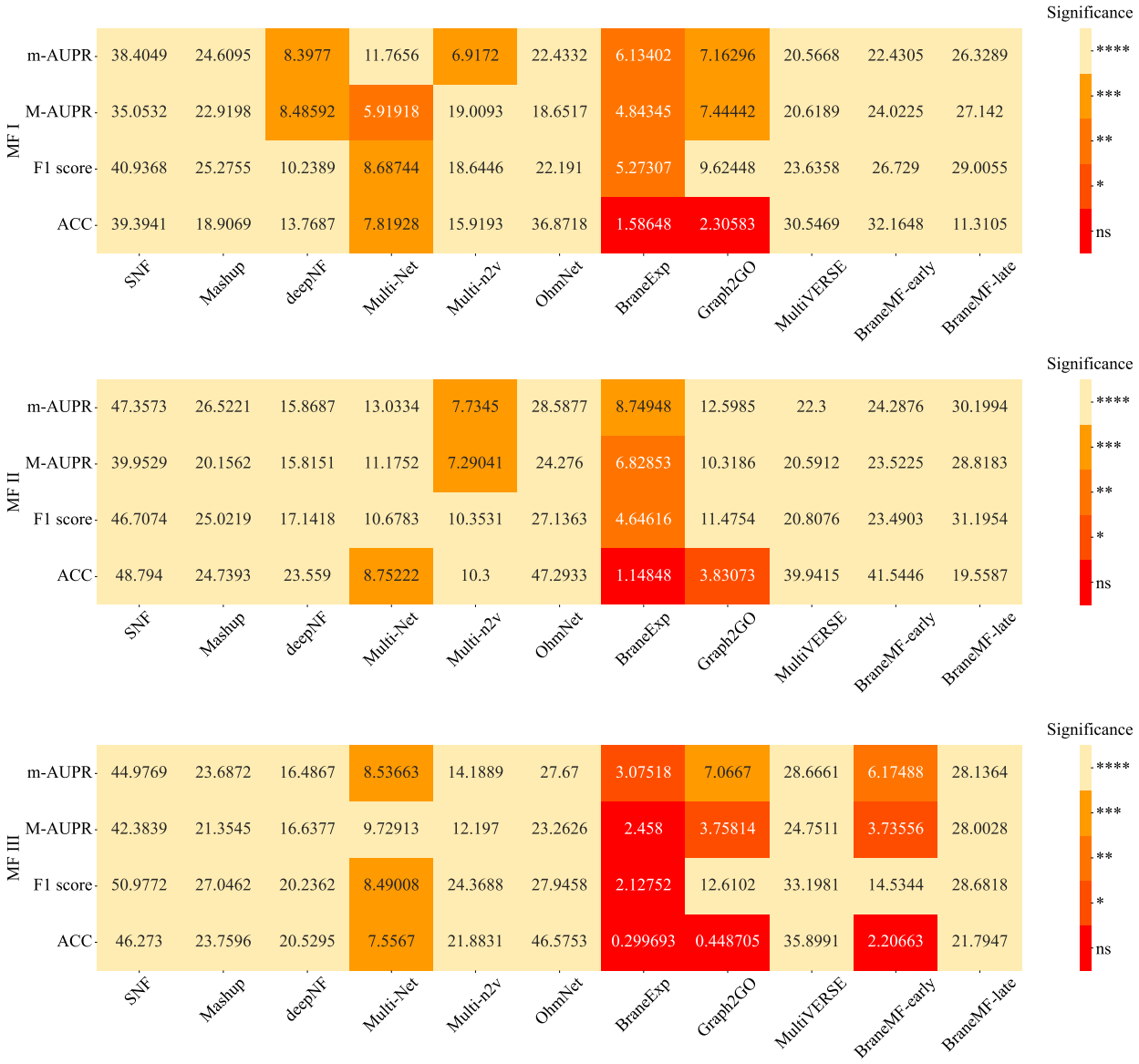
9. Statistical significance analysis

To determine whether the results are statistically significant compared to the baseline methods, we have applied a two sample t-test and calculated the p-values (P). These p-values show the probability of observing an effect of the same magnitude or more extreme given that the null hypothesis is true. A hypothesis test is designed to show whether the mean of the results of BraneMF is significantly different than the mean of the baseline method. We perform statistical test on results of each level of BP, MF, and CC for the following metrics: m-AUPR, M-AUPR, F1, and ACC. We compare the statistical significance of the BraneMF results compared to individual baseline methods. The resulting P values are then grouped into 'ns', '*', '**', '***', and '****' showing the level of significance. Their details are given below:

- $P > 5.00e-02$: 'ns'
- $1.00e-02 < P <= 5.00e-02$: '*'
- $1.00e-03 < P <= 1.00e-02$: '**'
- $1.00e-04 < P <= 1.00e-03$: '***'
- $P <= 1.00e-04$: '****'

The results are shown in Figure S8. For BP II, BP III, CC I, and CC II, BraneMF's results are statistically significant compared to all the baseline models. For BP I, BP II, MF I, MF II, except for accuracy, BraneMF's results are statistically significant compared to all the baseline models. Nevertheless, accuracy can be a useful measure if we have the same amount of samples per class but if we have an imbalanced set of samples accuracy is not that useful. Therefore, the remaining three metrics have more weight over accuracy. Note that, the statistical significance tests can only reveal the significance of differences in the mean score. However, to actually see if the mean scores of BraneMF are lower or higher than the baselines, please refer to Table S7, S8, and S9.


BP I
m-AUPR 45.8727 28.7863 17.0551 20.5932 18.918 27.5783 11.1042 10.2868 28.3633 10.2273 31.4019
M-AUPR 28.4284 20.0424 14.3986 15.5002 17.4721 17.8064 7.79667 9.83304 23.4913 6.98232 27.3795
F1 score 39.941 33.8278 17.4975 17.1025 20.8687 21.8165 12.401 14.97 23.4645 7.32966 36.0429
ACC 30.3883 25.3416 12.725 6.9812 15.2605 38.9018 2.06062 2.15209 31.0448 1.13853 27.5405
BP II
m-AUPR 41.2784 25.1538 13.7844 14.0984 6.453 26.322 8.70607 6.09842 24.2726 5.67653 28.9652
M-AUPR 38.1189 22.447 15.1042 13.8816 4.51313 26.6708 9.54171 6.1695 24.0421 3.36914 29.2238
F1 score 36.0687 30.2261 16.438 9.46247 6.97216 24.1308 10.0255 13.4978 22.032 4.58063 32.2659
ACC 24.1797 22.8933 19.3065 12.3765 13.7593 39.9136 1.82673 5.14495 32.8764 4.59108 23.938
BP III
m-AUPR 60.0749 25.7898 18.272 10.3207 5.85672 34.1506 14.8865 5.12057 30.324 9.69087 32.359
M-AUPR 52.4955 22.3284 17.8303 11.6771 6.77495 31.9319 10.3124 5.9053 31.1936 10.1606 32.152
F1 score 56.0528 32.0439 22.5158 3.38881 3.32201 27.6248 12.7786 13.4438 20.5521 18.9882 40.7232
ACC 53.7342 31.1523 23.6298 7.45511 5.83741 54.3714 6.76973 20.7582 42.4208 14.7876 25.5696
SNF Mashup deepNF Multi-Net Multi-n2v OhmNet BraneExp Graph2GO MultiVERSE BraneMF-early BraneMF-late
Significance
****
***
**
*
ns


MF I
m-AUPR 38.4049 24.6095 8.3977 11.7656 6.9172 22.4332 6.13402 7.16296 20.5668 22.4305 26.3289
M-AUPR 35.0532 22.9198 8.48592 5.91918 19.0093 18.6517 4.84345 7.44442 20.6189 24.0225 27.142
F1 score 40.9368 25.2755 10.2389 8.68744 18.6446 22.191 5.27307 9.62448 23.6358 26.729 29.0055
ACC 39.3941 18.9069 13.7687 7.81928 15.9193 36.8718 1.58648 2.30583 30.5469 32.1648 11.3105
MF II
m-AUPR 47.3573 26.5221 15.8687 13.0334 7.7345 28.5877 8.74948 12.5985 22.3 24.2876 30.1994
M-AUPR 39.9529 20.1562 15.8151 11.1752 7.29041 24.276 6.82853 10.3186 20.5912 23.5225 28.8183
F1 score 46.7074 25.0219 17.1418 10.6783 10.3531 27.1363 4.64616 11.4754 20.8076 23.4903 31.1954
ACC 48.794 24.7393 23.559 8.75222 10.3 47.2933 1.14848 3.83073 39.9415 41.5446 19.5587
MF III
m-AUPR 44.9769 23.6872 16.4867 8.53663 14.1889 27.67 3.07518 7.0667 28.6661 6.17488 28.1364
M-AUPR 42.3839 21.3545 16.6377 9.72913 12.197 23.2626 2.458 3.75814 24.7511 3.73556 28.0028
F1 score 50.9772 27.0462 20.2362 8.49008 24.3688 27.9458 2.12752 12.6102 33.1981 14.5344 28.6818
ACC 46.273 23.7596 20.5295 7.5567 21.8831 46.5753 0.299693 0.448705 35.8991 2.20663 21.7947
SNF
Mashup
deepNF
Multi-Net
Multi-n2v
OhmNet
BraneExp
Graph2GO
MultiVERSE
BraneMF-early
BraneMF-late
Significance
****
***
**
*
ns

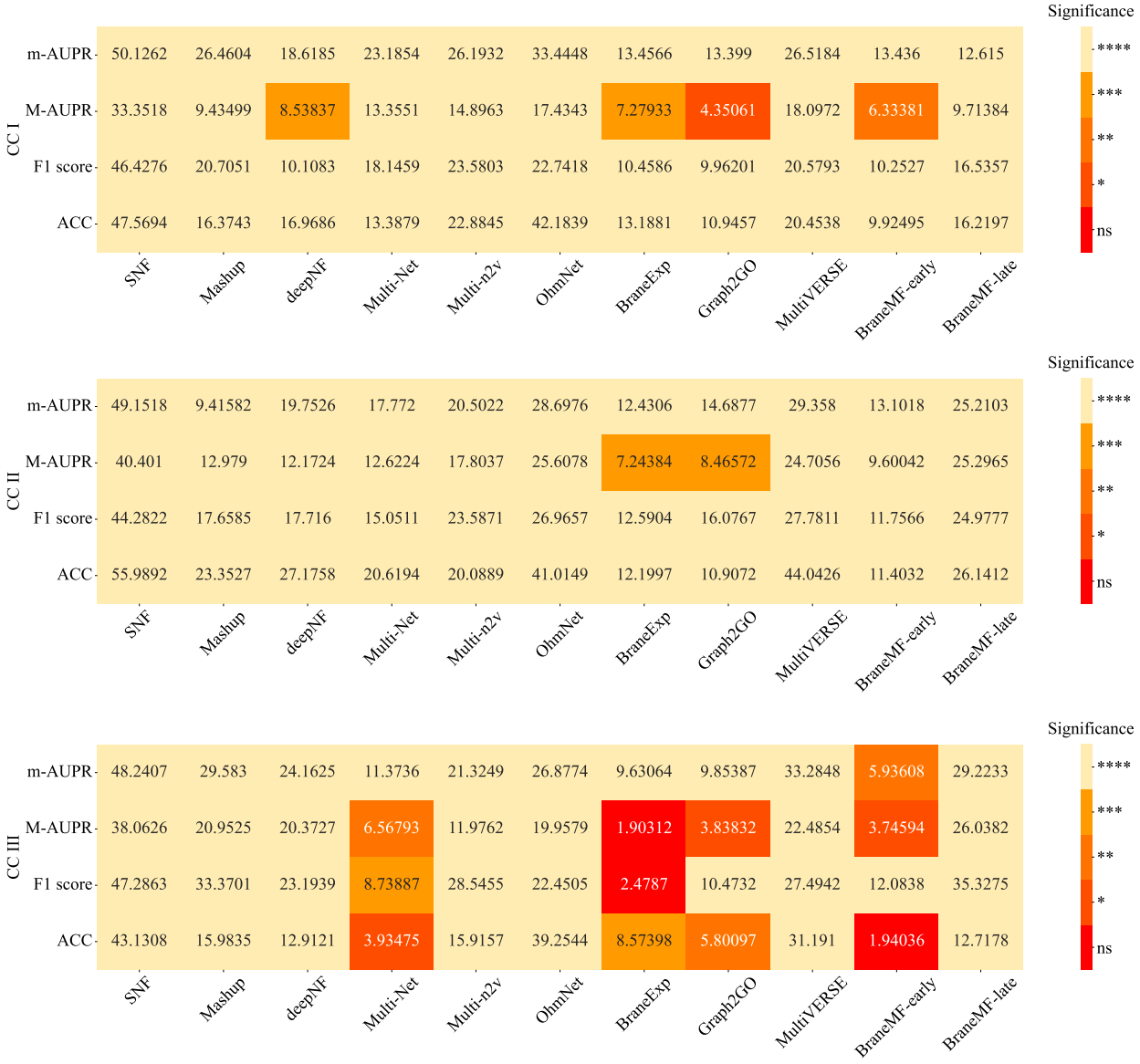


Figure S8: **Statistical significance of BraneMF compared to the baseline methods**. Each heatmap represents the statistical significance of protein function prediction for three levels of BP, MF, and CC measured by m-AUPR, M-AUPR, F1, and accuracy (ACC) scores. The values in the heatmap show the $-\log 10(p\text{-value})$ computed by applying a two-sided t-test for each metric respectively. The colors in the heatmap show the level of significance as described in Section 9.