Fast Embedding of Multilayer Networks: An Algorithm and Application to Group fMRI*

James D. Wilson†, Melanie Baybay‡, Rishi Sankar§, and Paul Stillman¶

September 19, 2018

Abstract

Learning interpretable features from complex multilayer networks is a challenging and important problem. The need for such representations is particularly evident in multilayer networks of the brain, where nodal characteristics may help model and differentiate regions of the brain according to individual, cognitive task, or disease. Motivated by this problem, we introduce the multi-node2vec algorithm, an efficient and scalable feature engineering method that automatically learns continuous node feature representations from multilayer networks. Multi-node2vec relies upon a second-order random walk sampling procedure that efficiently explores the inner- and intra- layer ties of the observed multilayer network is utilized to identify multilayer neighborhoods. Maximum likelihood estimators of the nodal features are identified through the use of the Skip-gram neural network model on the collection of sampled neighborhoods. We investigate the conditions under which multi-node2vec is an approximation of a closed-form matrix factorization problem. We demonstrate the efficacy of multi-node2vec on a multilayer functional brain network from resting state fMRI scans over a group of 74 healthy individuals. We find that multi-node2vec outperforms contemporary methods on complex networks, and that multi-node2vec identifies nodal characteristics that closely associate with the functional organization of the brain.

Keywords: node2vec, skip-gram, fMRI, neural network, word2vec

*JDW gratefully acknowledges support on this project by the National Science Foundation grant NSF DMS-1830547.

†Department of Mathematics and Statistics, University of San Francisco. San Francisco, CA 94117 jdwilson4@usfca.edu

‡Department of Computer Science, University of San Francisco. San Francisco, CA 94117 mbaybay@dons.usfca.edu

§Henry M. Gunn High School, Palo Alto, CA 94307 rishi.sankar@gmail.com

¶Department of Marketing, Yale School of Management. New Haven, CT 06511 paul.stillman@yale.edu

1 Introduction

Multilayer networks have been extensively used to model multi-modal relationships among interacting units of a complex system. Multilayer network models have revealed important insights in a variety of applications, including the modeling of multi-transit transportation systems [14; 55], the study of the effects of social interactions on economic exchange [19], and the analysis of the relationship between functional connectivity in the brain and cognition [6; 5; 4; 43]. In addition to the aforementioned applications, many problems necessitate the use of multilayer network models over traditional static network analyses. This is particularly apparent in our motivating application to network neuroscience. As brains are inherently multi-modal - varying across time, person, and cognitive task - multilayer networks provide information that cannot be inferred by static network models alone.

In this paper, we consider the problem of network embedding for multilayer networks. A multilayer network of length m is a collection of networks or graphs $\{G_1, \ldots, G_m\}$, where the graph G_ℓ models the relational structure of the ℓth layer of the network. Each layer $G_\ell = (V_\ell, W_\ell)$ is described by the vertex set V_ℓ that describes the units, or actors, of the layer, and the edge weights $W_\ell = \{w_\ell(u, v) : u, v \in V_\ell\}$ that describes the strength of relationship between the nodes. Layers may be viewed as ordered or unordered depending on the application; thus, we view dynamic networks as a special case of multilayer networks with layers ordered through time. Note that layers in the multilayer sequence may be heterogeneous across vertices, edges, and size. Denote the set of unique nodes in $\{G_1, \ldots, G_m\}$ by $\mathcal{N}$, and let $N = |\mathcal{N}|$ denote the number of nodes in that set. Throughout the remainder of this paper, to signify the unique node set $\mathcal{N}$ we represent multilayer networks with m layers and node set $\mathcal{N}$ as $\mathbf{G}^m_{\mathcal{N}}$.

Multilayer networks are inherently complex and high-dimensional. Without further assumptions on $\mathbf{G}^m_{\mathcal{N}}$, inference on $\mathcal{N}$ necessitates the modeling of N^2 (possibly dependent) edge variables, which is computationally challenging even for moderately sized N. In light of this challenge, the aim of the current work is to learn an interpretable low-dimensional feature representation of the nodes in a multilayer network. In particular, we seek a D-dimensional representation

$$\mathbf{F} : \mathcal{N} \to \mathbb{R}^D, \tag{1}$$

where $D << N$. The function $\mathbf{F}$ can be viewed as an $N \times D$ matrix whose rows $\{\mathbf{f}_v : v = 1, \ldots, N\}$ represent the feature space of each node in $\mathcal{N}$.

Our work is motivated by applications to functional magnetic resonance imaging (fMRI) over populations of individuals. In this setting, multilayer networks model the functional connections between regions of the brain across different individuals. We propose a fast and scalable algorithm, called *multi-node2vec*, that learns the maximum likelihood estimator for $\mathbf{F}$ from complex multilayer networks via the application of the Skip-gram neural network model to multilayer neighborhoods of nodes in $\mathcal{N}$. Multilayer neighborhoods are constructed from random walks over the multilayer network. These neighborhoods form what we call a *bag of nodes*, which acts as a collection of contexts for each node in the network. These nodal contexts (neighborhoods) are passed to the Skip-gram model, which constructs representative node vectors that form the rows of $\mathbf{F}$ via stochastic gradient descent. This process

is illustrated in Figure 1. We provide a detailed description of the algorithm in Section 2.

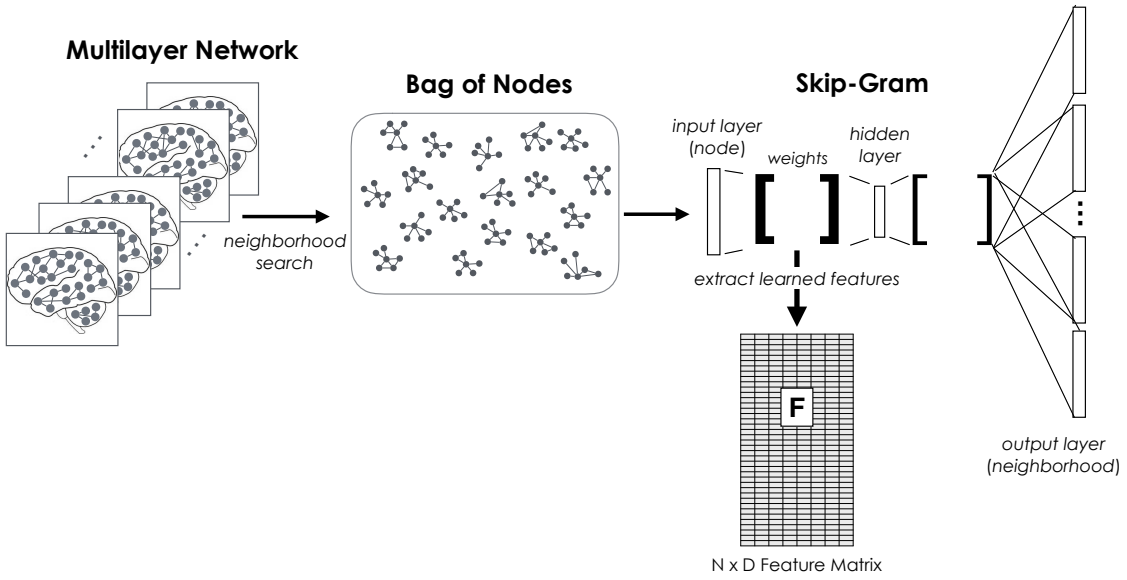

Figure 1: Demonstration of the multi-node2vec algorithm.

Our primary contributions are three-fold:

- We introduce multi-node2vec, a fast and scalable algorithm for embedding multilayer networks into lower dimensions. The algorithm utilizes the Skip-gram neural network model with negative sampling to perform maximum likelihood estimation of the nodal features given the observed multilayer network. The core of the algorithm relies on a suitable definition of a multilayer neighborhood, which we define as a collection of nodes that are likely to be visited from a random walk over the entire multilayer network. To the best of our knowledge, this is the first such algorithm to be designed for multilayer networks and can directly handle networks with heterogeneous layers.

- We derive the limiting behavior, in terms of the length of the random walk, of the multi-node2vec algorithm under which we derive an explicit relationship of the algorithm with matrix factorization of the $N \times N$ adjacency matrix characterizing an adjusted aggregate representation of the multilayer network. Furthermore, we establish a precise relationship of multi-node2vec with the node2vec and DeepWalk algorithms.

- We apply multi-node2vec to a multilayer brain network representing the functional connectivity of 74 healthy individuals who underwent resting state fMRI. In this case study, we demonstrate how to utilize the results of multi-node2vec for three primary objectives in the field of network neuroscience: (i) visualization of the unique nodes in the network once the network has been embedded onto D-dimensional space, (ii) clustering of the nodes into communities of similar features, and (iii) classification of nodes into anatomical regions of interest in the brain. We find that multi-node2vec identifies features that closely associate with the functional organization of the brain.

We also find that multi-node2vec significantly outperforms existing network embedding methods designed for single-layer networks.

The remainder of this manuscript is organized as follows. In Section 1.1 we provide an overview network neuroscience and discuss how the multi-node2vec algorithm has the potential to provide groundbreaking insights in the field. In Section 2 we discuss related work and other feature engineering techniques for complex networks. We describe the multi-node2vec algorithm in detail in Sections 3 and 4. We cast problem 1 as a maximum likelihood problem, and discuss how to identify multilayer neighborhoods in a multilayer network. We then study the asymptotic nature of the multi-node2vec algorithm and show that the algorithm is a fast approximation of a matrix factorization problem related to an aggregate representation of the network. We furthermore show a precise relationship of multi-node2vec with the node2vec and DeepWalk algorithms. In Section 6, we apply multi-node2vec to a resting state fMRI case study, and compare its performance with contemporary methods. We further investigate the performance of multi-node2vec on a test bed of simulated networks in Section 7. We conclude with a discussion of areas of future work in Section 8.

1.1 Motivation in Network Neuroscience

Multi-node2vec is motivated by the use of network analytic techniques to understand the interactions of the brain. This perspective, referred to as *network neuroscience*, views cognition as an emergent phenomenon of complex interactions amongst many different brain regions [10; 39; 50; 51]. The shift to studying (single-layer) brain networks has yielded many advances. For instance, network investigations of neural connectivity have revealed general organizing principles of the whole brain, including high modularity [52], a "rich-club" of interconnected hub regions [58; 57], and topologies that demonstrate small-world structure [2; 1; 3; 26]. These findings have shown, for instance, that the regions of the brain not only exhibit strong clustering, but also enable the brain to minimize wiring costs while maintaining robust transfer and integration of information across regions [13; 20]. Network investigations have also advanced our understanding of neural processes, such as learning and memory [4; 5], cognitive control [16], and emotion [29]. Network research has also demonstrated systematic network differences within the brains of individuals suffering neurological disorders such as concussion [36; 63; 62], Alzheimer's disease [17], and schizophrenia [22; 35; 59].

Brain networks are inherently multilayer – they vary across time, across person, and across cognitive task [6]. To date, however, a majority of network neuroscience strategies are static and consider only a single network of the brain. As a result, researchers who are interested in characterizing networks across (for instance) different people either analyze each individual network separately before comparing across analyses, or aggregate the connectivity across people [e.g., 54]. Such single-layer analyses neglect heterogeneity among individuals as well as their interdependencies [61]. Multilayer network representations of the brain enable researchers to fully analyze the relationships within and between networks observed over time, person, or task [e.g., 4; 5].

To construct a brain network, researchers typically divide the brain into a collection of regions, and then gauge the connection strength between regions. This is typically done via assessing structural connections (referred to as structural connectivity) between regions –

most commonly via assessing white matter tracts between regions – or functional coactivation of different regions [referred to as functional connectivity, 48; 50]. In resting state functional connectivity – the focus of the current study – researchers gauge the degree to which two regions' time-series activity is related to one another, with the logic being that greater two regions are functionally connected, the more their time-series' should appear to co-activate. In the present study, we model the strength of connection between two regions based on the correlation between the two regions' activity during a resting state fMRI scan [i.e., when participants have no task except to stay awake; 12; 49]. Given multiple people, time-points, or cognitive tasks, a multilayer network can be constructed so that each layer represents the strength of correlations between each pair of regions for a designated person. See Figure 2 for a demonstration of this construction.

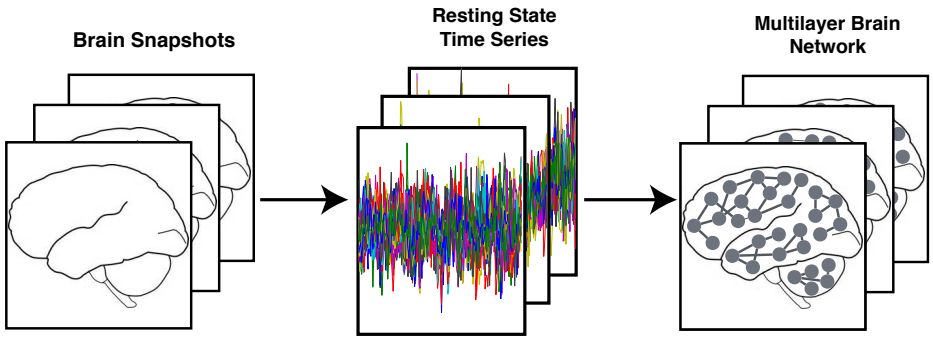

Figure 2: Illustration of the construction of multilayer brain networks.

Network neuroscientists have very recently begun to utilize multilayer analyses [6; 5; 4; 43; 9]. The majority of this work has explored how community structure and network modules vary across time. For instance, one study showed that shifts in community structure across time predict differences in learning a visual-motor task [5]. Recently, network neuroscientists have called for greater emphasis on multilayer techniques, particularly those that do not require temporal ordering of layers, thus allowing for more comprehensive quantification of networks across samples [43].

With multi-node2vec, we aim to provide neuroscientists with a scalable computational technique to automatically learn interpretable local features of the brain through the analysis of multilayer brain networks. In our initial demonstration in Section 4, we illustrate how multi-node2vec can be used to effectively segment regions into subnetworks with known cognitive functions. We furthermore demonstrate how to utilize the features identified by multi-node2vec to classify regions of the brain according to known clinical labels. Our novel feature engineering technique provides a valuable step in automatically learning neurological variation among brains, including individual differences and disease.

2 Related Work

Feature engineering is a common and important learning task in statistics and machine learning. Traditionally, feature engineering for networks, often referred to as network embedding, has amounted to manually describing summaries of networks based on a collection of user-selected network properties, like structural importance or subgraph counts [21; 27]. A similar strategy has been applied to multilayer networks, where chosen features attempt to quantify the within and between layer relationships among nodes [8; 31]. In contrast to these approaches, the multi-node2vec algorithm automatically learns important continuous features of multilayer networks and requires no user input on what properties to capture.

Feature engineering techniques have been extensively used to identify low-rank representations of multivariate data. In this setting, the data matrix $\mathbf{X}$ is an $n \times p$ matrix whose rows are independent observations measured on p features and. Dimension reduction techniques are particularly important when the data is high-dimensional – when $p > n$ – as traditional statistical inference is often no longer reliable [11]. Singular value decompositions, principal components analysis, and spectral clustering, for instance, are well-studied decomposition techniques that have been applied to a number of high-dimensional problems ranging from topic modeling to micro-array analysis. These techniques rely on the spectral decomposition of $\mathbf{X}$, its empirical covariance matrix, and the graph Laplacian of a similarity matrix on the columns of $\mathbf{X}$, respectively. Though these methods are known to provide accurate representations of $\mathbf{X}$, they face a drawback in computational complexity for large p due to matrix inversion, which can be prohibitive for especially high-dimensional problems. In Section 5, we show that multi-node2vec is in fact an approximation to closed-form implicit matrix factorization.

There have been many feature learning techniques for static networks developed in the past decade. The latent space model from [28], for example, is a common model-based embedding technique that embeds the observed network onto Euclidean space - typically onto two-dimensions. Our current work is most closely related to the automatic feature learning techniques LINE [56], DeepWalk [45], and node2vec [24]. We briefly discuss these here but refer the reader to [23] for a recent review of node embedding techniques for static networks. LINE, DeepWalk, and node2vec each learn features of a node from the neighborhoods of the node in the observed graph. LINE learns d-dimensional features by an objective function that preserves first and second order network properties. DeepWalk and node2vec each learn D-dimensional features using the Skip-gram neural network model, which minimizes a log-likelihood loss function that characterizes relationships from node neighborhoods in the observed graph. The Skip-gram model was originally developed for learning efficient representations of words in a large document of text [40; 44]. The first application of the Skip-gram model was in the word2vec algorithm [41], where it was used to estimate a word's features through the log-likelihood cost minimization from the prediction of that word's context. DeepWalk, node2vec, and multi-node2vec differ in the way they collect node neighborhoods. DeepWalk extracts neighborhoods using truncated random walks. Node2vec performs second random walks based on hyperparameters that guide the likelihood of visiting nodes either closer to or further away from previously visited nodes. Multi-node2vec is also random walk based and can be thought of as a generalization of the original DeepWalk and node2vec algorithms. Utilizing Laplacian dynamics like that discussed in [42], we incorporate a walk

parameter that dictates the probability of moving from one layer to the next.

Finally, the community detection task of partitioning the nodes of a multilayer network into densely connected subgroups, or communities, can be viewed as multilayer embedding. Specifically, the results of a community detection algorithm is an $N \times D$ matrix $\mathbf{F}$, where the vth row $\mathbf{f}_v$ is a binary vector that indicates which community(ies) the node v is contained. The development of multilayer community detection methods is still in its early stages, but several useful techniques have been developed over the past decade ([18; 42; 53; 61]). Though not the focus of this paper, it would be interesting to fully explore the use of communities as features for regression and other machine learning tasks in future work.

3 Maximum Likelihood Formulation

Let $\mathbf{G}_{\mathcal{N}}^m$ be an observed multilayer network with m layers and the set of unique nodes $\mathcal{N}$. Our aim is to learn D representative features of $\mathcal{N}$ given by the matrix $\mathbf{F}$ in (1). This learning task can be formulated as a problem of maximum likelihood estimation. To see this, one can view $\mathbf{G}_{\mathcal{N}}^m$ as a realization of a random graph on the node set $\mathcal{N}$ whose joint probability distribution is dictated by the feature matrix $\mathbf{F}$. Solving (1) is then equivalent to identifying the estimator $\widehat{\mathbf{F}}$ that maximizes the joint likelihood

$$\mathbb{L}(\mathbf{F} \mid \mathbf{G}_{\mathcal{N}}^m) = \mathbb{P}(\mathbf{G}_{\mathcal{N}}^m \mid \mathbf{F}), \tag{2}$$

where $\mathbb{P}$ is the joint distribution of a multilayer graph with m layers and unique node set $\mathcal{N}$ given the feature representation $\mathbf{F}$. In general, maximization of (2) is computationally intractable. We therefore make two simplifying assumptions about the joint distribution $\mathbb{P}(\cdot)$. Our assumptions rely upon a suitable definition of a *multilayer neighborhood*. Defining the neighborhood of a node is related to the problem of defining the context of a word in a large document from natural language processing. In static unweighted networks, the neighborhood of the node u is often defined as the collection of nodes that share an edge with u. This definition is motivated by the homophily principle [38], which posits that nodes with similar features are highly connected to one another in the network. In many cases this definition of a neighborhood is restrictive. This is particularly true when the observed network is only partially observed or when the edges of the network are generated from some underlying noisy process. We instead define a multilayer neighborhood of node u based on a dynamic process across the network. Our construction is a generalization of the random walk constructions from Grover and Leskovec [24]; Perozzi et al. [45]; Tang et al. [56] and is analogous to the defining of communities via Laplacian dynamics as in Lambiotte et al. [32]; Mucha et al. [42]. To be specific, we define the neighborhood of node u as the collection of vertices that are visited over a random walk on the multilayer network $\mathbf{G}_{\mathcal{N}}^m$. We make this more formal in Section 4.1 when describing the neighborhood search procedure of the algorithm.

Once the multilayer neighborhood of each node has been defined, we make two simplifying assumptions given the feature matrix $\mathbf{F}$. First, we assume that the joint distribution characterizing $\mathbf{G}_{\mathcal{N}}^m$ is the same as the distribution characterizing the collection of neighborhoods in $\mathbf{G}_{\mathcal{N}}^m$. This assumption is reasonable if we believe that the features $\mathbf{F}$ provide the

same information as the multilayer network itself. Second, given the feature matrix $\mathbf{F}$ we assume that the neighborhood of a node v depends only on its own feature representation, $\mathbf{f}_v$ and given this representation is independent of the neighborhoods of other nodes $u \in \mathcal{N}$. These assumptions are the same as those made for the node2vec algorithm for static networks [24] and are analagous to those made for word2vec, which assumes that the joint probability distribution of a collection of text can be characterized by the distribution of the collection of conditionally independent word contexts given each word's feature representation [40].

With the conditional independence assumptions of the neighborhoods given $\mathbf{F}$, maximizing the joint likelihood of $\mathbf{F}$ given the entire network $\mathbf{G}_{\mathcal{N}}^m$ reduces to the task of identifying the features $\mathbf{f}_v$ given the neighborhood of v in $\mathbf{G}_{\mathcal{N}}^m$. Given the neighborhood of each node, the likelihood from (2) simplifies to

$$\begin{aligned} \mathbb{L}(\mathbf{F} \mid \mathbf{G}_{\mathcal{N}}^m) &= \mathbb{P}\left(\bigcap_{u \in \mathcal{N}} \mathrm{Ne}(u) \mid \mathbf{f}_u\right) \\ &= \prod_{u \in \mathcal{N}} \mathbb{P}\left(\mathrm{Ne}(u) \mid \mathbf{f}_u\right). \end{aligned} \tag{3}$$

As it remains a challenging task to quantify the dependence between the neighborhoods of differing layers, the maximization of (3) is still computationally difficult. Thus, we define a family of multilayer graphs for which this maximization is feasible. It turns out that we can define such a family by assuming minimal conditional independence assumptions given the representation $\mathbf{F}$, described as follows. Let $\mathbf{G}_m(\mathcal{N})$ denote the family of multilayer graphs whose members are random graphs with m layers and unique nodes $\mathcal{N}$. For every member of $\mathbf{G}_m(\mathcal{N})$, assume that the following hold

(A1) For all $u \in \mathcal{N}$,

$$\mathbb{P}(\mathrm{Ne}(u) \mid \mathbf{f}_u) = \prod_{v \in \mathrm{Ne}(u)} \mathbb{P}(v \mid \mathbf{f}_u)$$

(A2) Let $u \in \mathcal{N}$. For every $v \in \mathrm{Ne}(u)$,

$$\mathbb{P}(v \mid \mathbf{f}_u) = \mathbb{P}(u \mid \mathbf{f}_v).$$

Assumption (A1) characterizes the local conditional independence among nodes in the neighborhoods of a node v given its feature representation, $\mathbf{f}_v$. Assumption (A2) enforces a symmetric effect of neighboring nodes in their feature space. A consequence of (A3) is that for any node v that is a neighbor of u, the following relationship holds

$$\mathbb{P}(v \mid \mathbf{f}_u) = \frac{\exp\{\mathbf{f}_v^T \mathbf{f}_u\}}{\sum_{w \in \mathcal{N}} \exp\{\mathbf{f}_w^T \mathbf{f}_u\}}.$$

If the observed graph $\mathbf{G}_{\mathcal{N}}^m$ is a realization of a multilayer random graph from the family $\mathcal{G}$ under which assumptions (A1), (A2), and (A3) hold, then (3) can be expressed as

$$\mathbb{L}(\mathbf{F} \mid \mathbf{G}_{\mathcal{N}}^{m}) = \prod_{u \in \mathcal{N}} \prod_{v \in \mathrm{Ne}(u)} \frac{\exp\{\mathbf{f}_v^T \mathbf{f}_u\}}{\sum_{w \in \mathcal{N}} \exp\{\mathbf{f}_w^T \mathbf{f}_u\}}. \tag{4}$$

Maximizing (4) is equivalent to maximizing the following log-likelihood

$$\mathcal{L}(\mathbf{F} \mid \mathbf{G}_{\mathcal{N}}^{m}) = \sum_{u \in \mathcal{N}} \sum_{v \in \mathrm{Ne}(u)} \left[\mathbf{f}_v^T \mathbf{f}_u - \log(Z_u)\right], \tag{5}$$

where $Z_u = \sum_{w \in \mathcal{N}} \exp\{\mathbf{f}_w^T \mathbf{f}_u\}$ is a normalization constant for the node u. Following the approach of [24; 40], we approximate Z_u using negative sampling. We note however that Markov chain Monte Carlo sampling methods could also be used to approximate Z_u as in [60]. The use of Skip-gram with negative sampling is appealing for two reasons: (i) the algorithm is fast and scalable to large multilayer networks, and (ii) the strategy is closely related to matrix factorization [34; 47] as we will see in Section 5.

4 The multi-node2vec Algorithm

We now describe multi-node2vec, a scalable algorithm for learning low-dimensional representations of complex multilayer networks. Given an observed multilayer network $\mathbf{G}_{\mathcal{N}}^{m}$, multi-node2vec is an approximate algorithm that estimates $\mathbf{F}$ through maximization of the log likelihood function in (5). The algorithm consists of two key steps. First, the **NeighborhoodSearch** procedure identifies a collection of s neighborhoods of length l for $\mathbf{G}_{\mathcal{N}}^{m}$ through second order random walks on the network. The **NeighborhoodSearch** procedure depends on three hyperparameters - p, q, and r - that dictate the exploration of the random walk away from the source node and the tendency to traverse layers. Once a collection of neighborhoods or *BagOfNodes* have been identified, the log-likelihood in (5) is optimized in the **Optimization** step using stochastic gradient descent on the two-layer Skip-gram neural network model of context size k. The result of the **Optimization** procedure is a D-dimensional feature representation $\mathbf{F}$. Pseudo-code for the multi-node2vec algorithm is provided below and Figure 1 provides an illustration.

Algorithm: multi-node2vec $(\mathbf{G}_{\mathcal{N}}^{m}, D, k, s, l, p, q, r)$

Input: Network $\mathbf{G}_{\mathcal{N}}^{m}$, dimension D, walk length l, samples s, context size k, walk parameters p, q, r

$BagOfNodes$ = **NeighborhoodSearch**$(\mathbf{G}_{\mathcal{N}}^{m}, s, l, p, q, r)$

$\mathbf{F}$ = **Optimization**$(D, k, BagOfNodes)$

Return $\mathbf{F}$

We describe the **NeighborhoodSearch** and **Optimization** procedures in more detail next.

4.1 The NeighborhoodSearch Procedure

Multi-node2vec begins by parsing a multilayer network into a collection of neighborhoods for each unique node in $\mathcal{N}$. The **NeighborhoodSearch** procedure identifies this collection of neighborhoods, or *BagofNodes*, using s truncated second order random walks of length l. Without loss of generality, suppose that node labels among layers are registered in the sense that node u in vertex set V_ℓ represents the same actor as node u in vertex set $V_{\ell'}$. To construct the random walk, we consider the collection of weights $\{w_{\ell,\ell'}(u,v) : \ell, \ell' \in 1,\ldots,m; u, v \in \mathcal{N}\}$, where $w_{\ell,\ell'}(u,v)$ defines the edge weight between node u from layer ℓ and node v from layer ℓ'. For each $u \in \mathcal{N}$, let $C_u \geqslant 0$ be a fixed constant. We set

$$w_{\ell,\ell'}(u,v) = \begin{cases} C_u & \text{if } \ell' \neq \ell, v = u \\ w_\ell(u,v) & \text{if } \ell' = \ell \\ 0 & \text{otherwise} \end{cases}$$

The above construction sets the edge weight between u and itself in a different layer, C_u, to be non-negative. In this way, C_u quantifies the inter-layer strength of relationship for node u to itself in the observed network. As an example, if the observed network is unweighted, it is natural to set $C_u \equiv 1$. In weighted networks with edge weights on the unit interval, one may set C_u to be the reciprocal multiplicity of the node u, or generally as some measure of similarity between the layers of the network. Throughout, we set C_u to 1 for all u unless otherwise specified.

For an observed multilayer network and its edge weights defined as above, the **NeighborhoodSearch** procedure identifies s neighborhoods using second order random walks over the nodes and layers of length ℓ, constructed as follows. Let u_i be the ith node visited by the random walk and ℓ_i the corresponding layer. Suppose, without loss of generality, that the initial pair (u_1, ℓ_1) is chosen uniformly at random. Subsequent vertex, layer pairs are visited according to the conditional probability

$$\mathbb{P}(u_i = x, \ell_i = \ell' \mid u_{i-1} = v, \ell_{i-1} = \ell) = \frac{\pi_{v,x,\ell,\ell'}}{Z}, \;\; w_{\ell,\ell'}(v,x) > 0 \tag{6}$$

where $\pi_{v,x,\ell,\ell'}$ is the unnormalized transition probability of moving from vertex-layer pair (v,ℓ) to pair (x,ℓ'), and Z is a normalizing constant. We set $\pi_{v,x,\ell,\ell'}$ as a function of the walk parameters p, q, and r as follows

$$\pi_{v,x,\ell,\ell'} = \alpha_{pqr}(t,x,\ell,\ell') \cdot w_{\ell,\ell'}(v,x). \tag{7}$$

The $\alpha_{pqr}(t,x,\ell,\ell')$ term acts as a search bias on the observed weights that depends on the previously traversed edge (t,v). That is, the walk now resides at node v having just traveled from node t and the next node that the random walk visits depends on (a) the distance t is from the future node, and (b) whether there is a layer transition. Let $d_\ell(t,x)$ denote the shortest path distance between nodes t and x in layer ℓ. To account for layer transitions, we further decompose $\alpha_{pqr}(t,v,x,\ell,\ell')$ as

$$\alpha_{pqr}(t,x,\ell,\ell') = \beta_{pq}(t,x)\; \mathbb{I}(\ell' = \ell) + \gamma_r(v,x)\; \mathbb{I}(\ell' \neq \ell), \tag{8}$$

where

$$\beta_{pq}(t,x) = \begin{cases} p^{-1} & d_\ell(t,x) = 0 \\ 1 & d_\ell(t,x) = 1 \\ q^{-1} & d_\ell(t,x) = 2 \end{cases}$$

and $\gamma_r(v,x) = r^{-1}\,\mathbb{I}(x = v)$. The $\beta_{pq}(t,x)$ term controls the rate at which the random walk explores and leaves the neighborhood of a node within layer ℓ. This quantity is the same as that specified for static networks in node2vec and has been shown to identify neighborhoods that interpolate between outcomes of breadth first search and depth first search. The return parameter p controls the likelihood of revisiting the same node, layer pair; whereas, the in-out parameter q controls exploration of the walk in layer ℓ. The $\gamma_r(v,x)$ term controls the rate at which a random walk transitions from one layer to another. Setting the layer walk parameter r to be large ($> max(p,q,1)$) ensures little layer-to-layer exploration. Setting r in this way encourages independent neighborhood sampling across layers. On the other hand, setting r to be small ($< min(p,q,1)$) promotes exploration among layers, and the resulting neighborhoods will reflect dependency among the layers.

Once the parameters s, l, p, q, and r have been chosen, s random walks of length l are performed on the nodes of the observed multilayer network using transition probabilities from (6). These s samples serve as the *BagofNodes* from which the nodal features are learned.

4.2 The Optimization Procedure

For a given dimension size D, a context size k, and the collection of neighborhoods from the **NeighborhoodSearch** step, multi-node2vec then minimizes the cost of (5) using stochastic gradient descent and the Skip-gram two-layer neural network model. For each node the normalization constant Z_u is approximated using negative sampling. The Skip-gram model iteratively updates the matrix $\mathbf{F}$ in the following manner. Each node is encoded as a one-hot vector and provided as the input layer to a 2-layer neural network from which the neighborhood of the node is predicted. Applying the log-likelihood $\mathcal{L}$ as a cost function, the error of the prediction is calculated. Partial derivatives of the cost function with respect to the rows of each of the intermediate weight matrices are calculated and updated using stochastic gradient descent to minimize cost. This procedure is repeated across all nodes in $\mathcal{N}$ until the cost function can no longer be reduced. After learning from each of the neighborhoods in our bag of nodes, we extract the models *node embeddings* - the $N \times D$ representation weight matrix associated with Skip-gram's input layer.

This optimization is analogous to that of the node2vec algorithm, but in our application the weight matrices of the two layer neural network are D-dimensional representations of the unique nodes $\mathcal{N}$ and thus account for the dependendence among layers in the multilayer network. It should be noted that multi-node2vec is an approximate algorithm that relies upon the normalizing constants $\{Z_u\}$, as well as the approximate optimization of stochastic gradient descent. Though not the focus of this paper, there has been a lot of recent work investigating the optimality landscape of gradient descent methods (see for example [33]), which provides promising theoretical justification for its use.

The choice of k directly affects the amount of information one gains for each node but its value depends on the sparsity of the observed network. Large values of k introduce

undesired noise to the identified neighborhoods; whereas, values of k that are too small result in neighborhoods that do not contain significant information about the neighborhoods in the network. We found that setting k near the average degree of the network provided the best results in our numerical studies. In the case that the observed network is either densely connected or contains few layers, the neighborhoods for each node may not contain sufficient information to inform the desired features. In such scenarios, it may be desirable to sample multiple neighborhoods for each node. Thus, we include an optional parameter a that specifies the minimum number of samples generated for each node. Unless otherwise specified, we set $a = 1$ in our numerical studies. Finally, the dimensionality parameter D should be chosen to provide sufficient information about the multilayer network while greatly reducing the total number of nodes N, though it is an open problem to understand an optimal dimension to represent general static networks. We investigate the effects of context size and dimensionality choice in our numerical studies in Sections 6 and 7.

5 Relationship with node2vec, DeepWalk, and Matrix Factorization

Multi-node2vec is an approximate algorithm that seeks to maximize the log-likelihood objective function given in equation (5). Approximation is needed for two objectives - (i) the identification of multilayer neighborhoods via random walks, and (ii) the application of the Skip-gram neural network model with negative sampling. For (i), the analysis of multi-node2vec's transition probabilities in (6) reveals that multi-node2vec is in fact stochastically equivalent to running node2vec and DeepWalk on an aggregate representation of $\mathbf{G}_{\mathcal{N}}^{m}$. For (ii), by analyzing the asymptotic nature of the random walks in the **NeighborhoodSearch** procedure as $l \to \infty$, one can leverage the recent work on the Skip-gram with negative sampling from [34; 47] to show that multi-node2vec approximates implicit matrix factorization. We describe these main results below in turn.

5.1 Stochastic Equivalence with node2vec and DeepWalk

We begin by analyzing the relationship of multi-node2vec with the node2vec and DeepWalk algorithms. To do so, we need some preliminary notation. Suppose that the edge weights of the multilayer network $\mathbf{G}_{\mathcal{N}}^{m}$ are non-negative, and suppose that each layer is undirected. Let $\mathbb{A}$ denote the $N \times N$ aggregate adjacency matrix of the nodes $\mathcal{N}$ with entries $\mathbb{A}_{u,v} = \sum_{\ell} \sum_{\ell'} w_{\ell,\ell'}(u,v)$. Denote the multiplicity of the node u in $\mathbf{G}_{\mathcal{N}}^{m}$ by m_u, namely the number of layers in which the node u is contained. Define the adjusted version of $\mathbb{A}$, $\widetilde{\mathbb{A}}(r)$, as the $N \times N$ matrix with entries

$$\widetilde{\mathbb{A}}_{u,v}(r) = \begin{cases} \mathbb{A}_{u,v} & \text{if } v \neq u \\ r^{-1} m (m_u - 1) C_u + \sum_{\ell} w_{\ell}(u,u) & \text{if } v = u. \end{cases}$$

Note that $\widetilde{\mathbb{A}}(r) = \mathbb{A}$ when $r = 1$. One can view the matrix $\widetilde{\mathbb{A}}(r)$ as an adjusted adjacency matrix whose self-loop weights depend on the layer walk parameter r. Write $\widetilde{\mathbf{G}}_{\mathcal{N}}(r)$ as the

graph with nodes $\mathcal{N}$ and edge weights specified by the adjacency matrix $\widetilde{\mathbb{A}}(r)$.

To state our result, we need a notion of equivalence between two stochastic algorithms. For this purpose, we consider the stochastic equivalence of two algorithms, defined as follows.

Definition 1. *Let A_1 and A_2 be two stochastic algorithms, each with the same set of possible outcomes Ω. That is, for fixed input data X, A_k is a random function that maps X to an outcome $o \in \Omega$: $A_k(X) \to o \in \Omega$. Define $\mathbb{P}_k$ as the probability mass function characterizing the probability of each possible outcome of A_k: $\{\mathbb{P}_k(A_k(X) = o) : o \in \Omega\}$. A_1 and A_2 are said to be* **stochastically equivalent** *if $\mathbb{P}_1 = \mathbb{P}_2$.*

The following theorem relates multi-node2vec with node2vec and DeepWalk and shows under what conditions they are stochastically equivalent in terms of the walk parameters p, q, and r.

Theorem 2. *Let $\mathbf{G}_{\mathcal{N}}^m$ be an observed multilayer network and let $\widetilde{\mathbf{G}}_{\mathcal{N}}(r)$ be its adjusted aggregate network. Suppose that the parameters D, k, s, l are held constant. Then the following hold*

(a) *For all $p, q, r > 0$, the application of multi-node2vec to $\mathbf{G}_{\mathcal{N}}^m$ is stochastically equivalent to the application of node2vec to $\widetilde{\mathbf{G}}_{\mathcal{N}}(r)$.*

(b) *If $p = q = 1$, the application of multi-node2vec to $\mathbf{G}_{\mathcal{N}}^m$ is stochastically equivalent to the application of DeepWalk to $\widetilde{\mathbf{G}}_{\mathcal{N}}(r)$.*

We prove Theorem 2 in the Appendix. Theorem 2 reveals that the application of multi-node2vec on an observed multilayer network $\mathbf{G}_{\mathcal{N}}^m$ is stochastically equivalent to the application of node2vec on the adjusted aggregate graph $\widetilde{\mathbf{G}}_{\mathcal{N}}(r)$. In practice, this means that running multi-node2vec on an observed multilayer network will provide the same results as running node2vec on the corresponding adjusted aggregate network if the same seed set is specified for a random number generator. In the special case that $p = q = 1$, one can equivalently run multi-node2vec, node2vec, or DeepWalk.

5.2 Limiting Behavior as Matrix Factorization

Our next analysis relies upon the investigation of the Skip-gram neural network model with negative sampling from [34]. Denote $\mathcal{D}$ as the collection of neighborhoods identified by the **NeighborhoodSearch** procedure. Let $w = \{u_1, \ldots, u_l\} \in \mathcal{D}$ be a collection of nodes resulting from a length l random walk in the **NeighborhoodSearch** procedure. Define the k-length contexts for node u_i as the nodes neighborhing it in a k-sized window $u_{i-k}, \ldots, u_{i-1}, u_{i+1}, \ldots, u_{i+k}$ and let c denote the collection of contexts for w. Let $\#(w, c)$ denote the number of times the node-context pair (w, c) appears in $\mathcal{D}$. Further, let $\#(w)$ and $\#(c)$ denote the number of times the node w and the context c appear in $\mathcal{D}$, respectively. As shown in [34], running Skip-gram with negative sampling is equivalent to implicitly factorizing

$$\log\left(\frac{\#(w,c)|\mathcal{D}|}{\#(w)\#(c)}\right) - \log(b), \tag{9}$$

where b is the number of negative samples specified. Expression (9) suggests that by getting a hold of the quantity in the first logarithm of the expression, we can relate multi-node2vec directly to matrix factorization.

Our results provide asymptotic expressions for $\#(w,c)|\mathcal{D}|/\#(w)\#(c)$ when the random walk length $l \to \infty$. To make our result explicit, we need to first introduce a little more notation. Define $\widetilde{\mathbf{d}}_u = \sum_{v \in \mathcal{N}} \widetilde{A}_{u,v}(r)$ as the generalized degree of node u in $\widetilde{\mathbf{G}}_{\mathcal{N}}(r)$ and let $\widetilde{\mathbf{D}} = \text{diag}(\widetilde{\mathbf{d}}_1, \ldots, \widetilde{\mathbf{d}}_N)$. Define the volume of $\mathbf{G}_{\mathcal{N}}(r)$ as $\text{vol}(\widetilde{\mathbf{G}}_{\mathcal{N}}(r)) = \sum_{u \in \mathcal{N}} \widetilde{\mathbf{d}}_u$. Define $\underline{\mathbf{P}}$ as the array containing the second order transition probabilities of **NeighborhoodSearch**: $\underline{\mathbf{P}} = \{\underline{P}_{u,v,w} = P(u_{j+1} = u \mid u_j = v, u_{j-1} = w)\}$ and let $\mathbf{X}$ be its corresponding stationary distribution satisfying $\sum_w \underline{P}_{u,v,w} X_{v,w} = X_{u,v}$. Furthermore, let $\underline{P}^k_{u,v,w} = P(u_{j+r} = u \mid u_j = v, u_{j-1} = w)\}$ denote the kth step transition probability.

Finally, suppose $\xrightarrow{P}$ denotes convergence in probability. Our analysis of multi-node2vec depend on the bias of the transition probabilities for the random walks of the **NeighborhoodSearch** procedure in equation (7), $\alpha_{pqr}(t, x, \ell, \ell')$. We can now state our next theorem, which relates multi-node2vec directly with matrix factorization.

Theorem 3. *Let $\mathbf{G}^m_{\mathcal{N}}$ be an observed multilayer network and let $\mathbf{G}_{\mathcal{N}}(r)$ be its adjusted aggregate network. Suppose that $\mathbf{G}_{\mathcal{N}}(r)$ is connected, undirected, and non-bipartite. Let k be the context size chosen for the* **Optimization** *procedure. Then as $l \to \infty$,*

(a) *For all $p, q, r > 0$,*

$$\frac{\#(w,c)|\mathcal{D}|}{\#(w)\#(c)} \xrightarrow{P} \frac{1}{2k} \frac{\sum_{j=1}^{k} \left(\sum_u X_{w,u} \underline{P}^j_{c,w,u} + \sum_u X_{c,u} \underline{P}^j_{w,c,u} \right)}{\left(\sum_u X_{w,u} \right) \left(\sum_u X_{c,u} \right)} \tag{10}$$

(b) *Let $\widetilde{\mathbf{P}} = \widetilde{\mathbf{D}}^{-1} \widetilde{\mathbb{A}}$. If $p = q = 1$,*

$$\frac{\#(w,c)|\mathcal{D}|}{\#(w)\#(c)} \xrightarrow{P} \frac{vol(\widetilde{\mathbf{G}}_{\mathcal{N}}(r))}{k} \left(\sum_{x=1}^{k} \widetilde{\mathbf{P}}^k \right) \widetilde{\mathbf{D}}^{-1} \tag{11}$$

for all $r > 0$.

We prove Theorem 3 in the Appendix. Results (10) and (11) provide closed form limiting expressions for the matrix factorization problem in (9). These results suggest the use of matrix factorization to identify features for a multilayer network; however, it should be noted that calculating and storing the second order transition probabilities $\underline{\mathbf{P}}$ and its stationary distribution $\mathbf{X}$ is computationally prohibitive. We do not consider such an algorithm in our current study but plan to address fast matrix factorization in future work.

6 Case Study: Multilayer Brain Networks for Resting State fMRI

We now apply multi-node2vec to a multilayer brain network representing the functional connectivity of 74 healthy individuals who underwent resting state fMRI. In this case study,

we test the utility of multi-node2vec for three primary objectives: (i) visualization of the unique nodes in the network once the network has been embedded onto D-dimensional space, (ii) clustering of the nodes into communities of similar features, and (iii) classification of nodes into anatomical regions of interest in the brain. To assess overall performance, we compared multi-node2vec with several off-the-shelf embedding techniques, including LINE, DeepWalk, node2vec, and the spectral decomposition of the graph Laplacian. Our analysis reveals that multi-node2vec identifies features that closely associate with the functional organization of the brain. Furthermore, we find that multi-node2vec is comparable to off-the-shelf embedding methods when the observed multilayer network is homogeneous throughout its layers, and significantly outperforms its competitors when the observed multilayer network contains noisy layers.

Description of Data

We investigate a data set of resting-state fMRI scans of 74 healthy individuals (ages 18-65, 23 female) from the Center for Biomedical Research Excellence [COBRE, 37] posted to the 1000 Functional Connectomes Project [7]. Participants had no history of neurological disorder, mental retardation, substance abuse or dependencies in the last 12 months, or severe head trauma. Participants underwent 5 minutes of resting state fMRI in which they had no task except to stay awake, followed by a multi-echo MPRAGE scan (see the supplement for scanning parameters and preprocessing information).

To construct the multilayer representation of this data set, we use a previously validated atlas [46] that specifies 264 spheres of radius 8mm, which constitute our 264 regions of interest (ROIs). We averaged the fMRI time-series' from all voxels within each ROI, yielding 264 time-series' per participant. For each of these time series', we regressed out 6 motion parameters (to account for head movement), 4 parameters corresponding to cerebrospinal fluid, and 4 parameters corresponding to white matter. These steps have been shown to reduce bias and noise within the data [15]. Finally, for each participant, we correlated the 264 time-series' with one another, yielding a 264 * 264 correlation matrix for each participant. We then thresholded the correlation matrices with a threshold of 0.10 (i.e., retaining the top 10% strongest connections) to construct the multilayer network (of 74 layers and 264 unique nodes) for which we apply multi-node2vec. The functional atlas further provides each ROI (node) in the network with a label that specifies the functional subnetwork in which the ROI is contained [46]. We provide a summary of the subnetwork labels in Table 1.

We investigate the utility of multi-node2vec in visualization as well as the machine learning tasks of clustering and classification, using the subnetwork labels of the ROIs as the ground truth. Publicly available code for the multi-node2vec algorithm as well as all code used for the findings in this section as well as Section 7 are available in the supplemental material.

6.1 Visualizing and Clustering Regions of Interest

For all uses of multi-node2vec, we set k = 10 and D = 100 to produce feature matrices of dimension 264 x 100. We sampled s = 52 neighborhoods for each node. These parameter settings were chosen based on the performance of multi-node2vec on a test bed of simulated networks with similar size and density to the brain network as described in Section 7. We set p = 1, and q = 0.5 to match the parameter settings of node2vec as suggested in Grover

Table 1: Summary of subnetwork labels in the fMRI multilayer brain network.

Auditory	*Dorsal Attention*	*Cingulo-opercular Task Control*	*Default Mode*
13	11	14	58
Salience	*Memory/retrieval*	*Fronto-parietal Task Control*	*Sensory/somatomotor – Hand*
18	5	25	30
Visual	*Ventral Attention*	*Subcortical*	*Sensory/somatomotor – Mouth*
31	9	13	5
Cerebellar	*Uncertain*		
4	28		

and Leskovec [24], and we ran multi-node2vec across $r = 0.25$, 0.50, and 0.75.

We first assessed the relevance of the features identified by multi-node2vec through evaluating the extent to which the nodes (rows) of the feature matrices clustered into groups that match the ground truth subnetwork labels. We ran multi-node2vec with layer walk parameter $r = 0.25, 0.50$, and 0.75 on the multilayer imaging network embedding the network onto $D = 100$ features. As a first step, we explore the pairwise scatterplots of pairs of the features to investigate whether clustering is present among ROIs of similar function. Figure 3 shows the pairwise scatterplots of features 7, 8, and 9 when $r = 0.25$. We see clear clustering among nodes of differing functional subnetworks, particularly along features 7 and 8. These three features were chosen visually, but they highlight the potential use of multi-node2vec's output in distinguishing subnetworks of the brain.

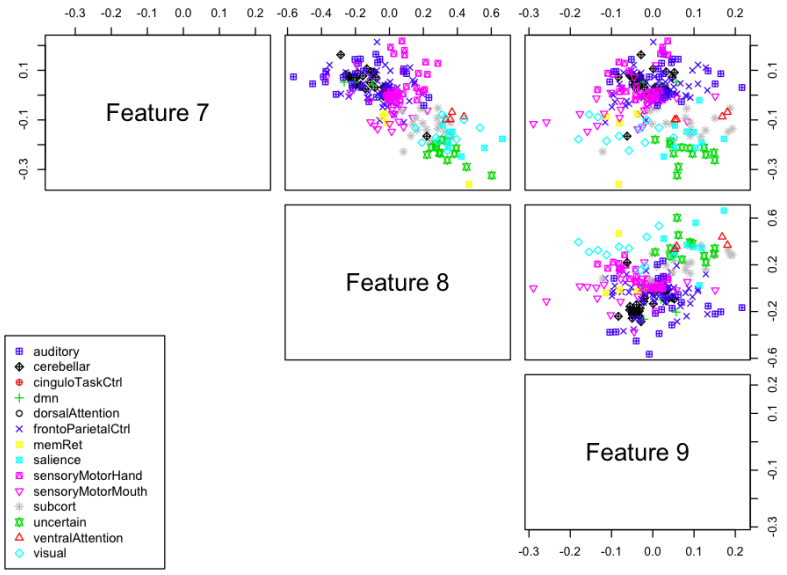

Figure 3: Pairwise scatterplots of the 7th, 8th, and 9th node embeddings from multi-node2vec with layer walk parameter $r = 0.25$. This example highlights the potential use of node embedding to better visualize distinct functional subgraphs.

To explore functional region segmentation further, we next clustered the rows of the feature matrices identified from multi-node2vec across all three walk parameter settings. For

this task, we were particularly interested in the effect of the feature dimension on clustering performance. To test this effect, we proceeded as follows. The first D of 100 features (columns) were held out from each identified feature matrix. The k-means clustering algorithm was then applied on the rows of the resulting $N \times D$ matrix, and the number of clusters was set to 13 to match the true number of subnetwork labels. For each run, the identified clusters were compared against the true subnetwork labels using the adjusted rand score. We repeated this process for each method across a grid of D from 2 to 100 in increments of 2. The results for each method are plotted in Figure 4.

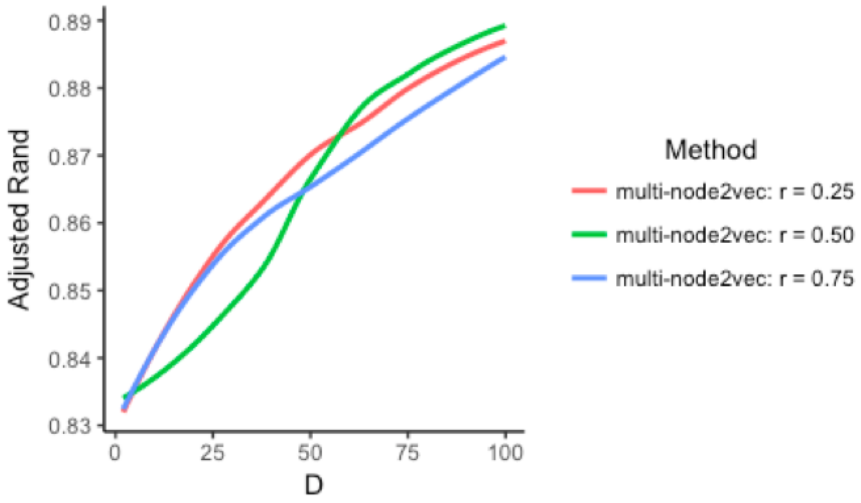

Figure 4: The adjusted rand score of the best 13 clusters identified using k-means on the observations (rows) of the feature matrices identified by multi-node2vec with D of 100 total features.

Figure 4 reveals that the features identified by multi-node2vec provide biologically relevant information about the function of regions in the brain. The match of the identified clusters with the ground truth improves as the number of features, D increases. Notably, even for D as small as 5, the ROI clusters closely resemble the ground truth labels (adjusted rand ≈ 0.83).

We note that plots like the one given in Figure 4 provide a heuristic for assessing how many dimensions should be used to capture a desired ground truth in a multilayer network. For example, in this case we can use even just 2 dimensions and still capture more than 80% of the ground truth in functional subnetworks. These visualization and clustering results reveal that the features of multi-node2vec provide practically relevant information about the functional subnetwork to which these ROIs belong. This finding is further supported in the classification study considered performed next.

6.2 Classification of Functional Subnetworks

We now assess the utility of the features learned from multi-node2vec through the classification task of predicting the subnetwork location for each ROI. We considered the classification of the nine subnetworks containing ten or more ROIs, which included the *au-*

ditory, cingulo-opercular task control, default mode, fronto-parietal task control, salience, sensory/somatomotor – hand, subcortical, visual, and *dorsal attention* subnetworks. In the classification task, we tested two scenarios for network embedding methods – (i) the multilayer network representing the resting state fMRI of 74 healthy individuals alone, and (ii) the multilayer network with additional noisy layers.

For each subnetwork, we trained a one-versus-all logistic regression classifier on the rows of the feature matrix for each method on 80% of the regions using the first D identified features. We applied the classifier to the remaining 20% of the ROIs and assessed the performance of the classifier using the area under the curve (AUC). We performed this classification on the feature matrices for each method and calculated the resulting AUC of the classifier across D ranging from 2 to 100 in increments of 2.

For comparison, we also applied node2vec, DeepWalk, LINE and the spectral decomposition to the average thresholded network. For node2vec, we set the return parameter as p = 1 and the in-out parameter as q = 0.5 to guide the neighborhood search following the suggestions of the original paper. For DeepWalk, we kept default parameters. Matching multi-node2vec, we set k = 10 for both node2vec and DeepWalk. For LINE, we used its default parameters: negative-sampling = 5 and $\rho = 0.025$. To match LINE's default of 1 million training samples, we sampled s = 3,788 neighborhoods for each node in node2vec and DeepWalk. For spectral, we calculated the eigenvectors of the normalized graph Laplacian of the aggregate matrix. We ran all methods to learn D = 100, thus each identified a 264 x 100 feature matrix. All experiments were performed on an AWS T2.Xlarge instance (specs: a 64-bit Linux platform with 16 GiB memory). We report the AUC for each method and each subnetwork in Figures 5, 6, and 7.

Our study reveals that multi-node2vec is comparable to the competing methods in the non-noisy setting, where we expect layers to be homogeneous across the healthy patients. We further find that multi-node2vec is robust to multilayer networks with additional noisy layers. Indeed in this setting, we find that multi-node2vec outperforms its competitors in seven of nine classification studies. These results provide evidence of the robustness of multi-node2vec across multilayer networks with heterogeneous layers and reveal the overall utility of the algorithm for noisy and non-noisy networks. We discuss these results in more detail next.

We begin by analyzing the classification result on the original 74 individuals, presented in Figure 5. Since each individual in the original study is healthy, we expect the networks of each these individuals to share similar structure. It follows that the aggregate network provides an unbiased summary of the multilayer network with less variability than each layer alone. Thus methods applied to the aggregate network are expected to do better than multi-node2vec. Despite this, we see from Figure 5 that multi-node2vec is comparable to the competing methods for seven out of nine subnetworks and outperforms other methods for small D in the *visual* and *sensory-motor (hand)* regions. The LINE method does particularly well in the *salience* and *dorsal attention* classifications, and outperforms multi-node2vec and all other methods across D. All methods improve with increasing D and approach 1, indicating perfect classification.

To test the performance of multi-node2vec on multilayer networks with noise, we next generated b layers, each with 264 nodes to match the number of regions in every other layer, from an Erdős-Rényi with edge probability set to the average edge density across all 74

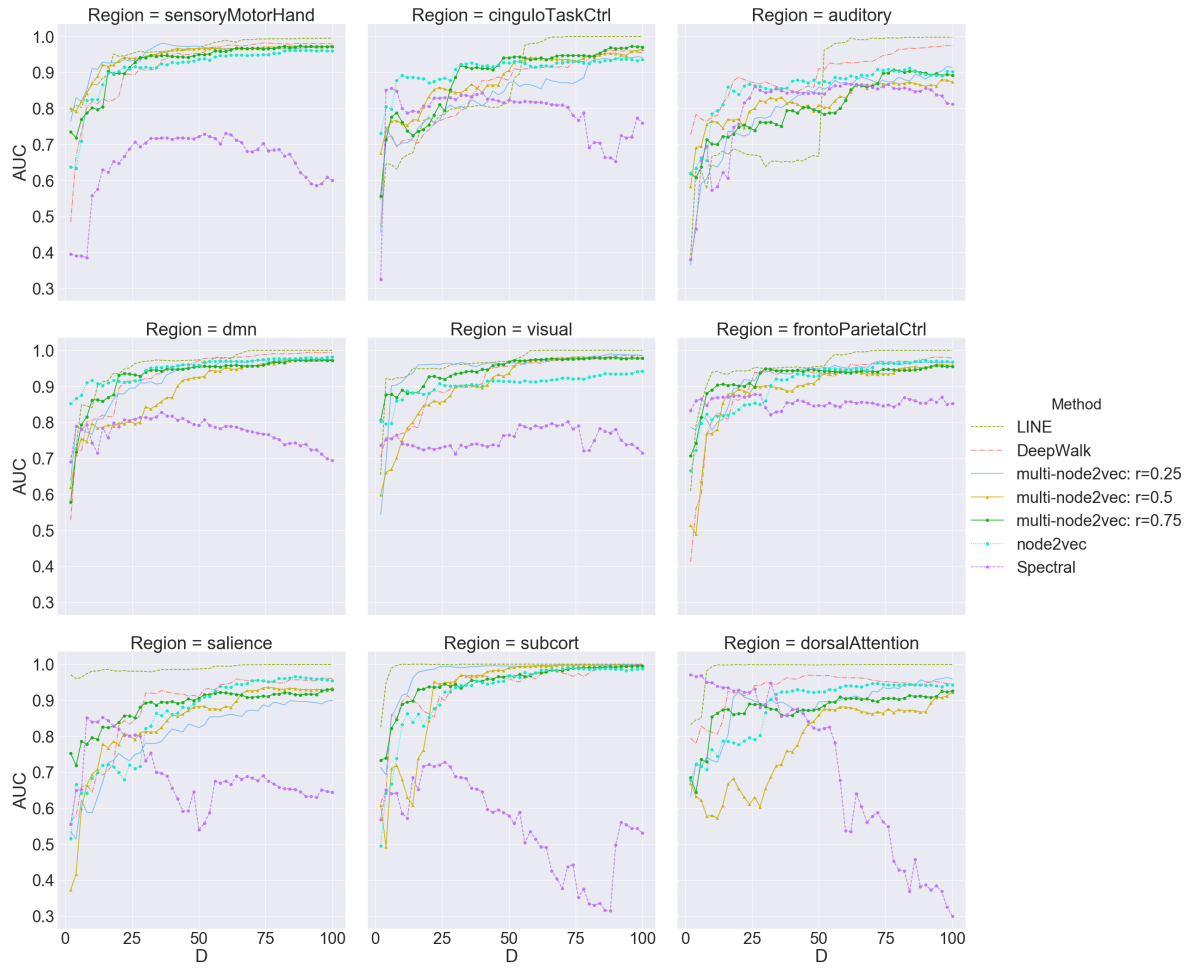

Figure 5: The AUC of a one vs. all logistic regression classifier for the nine major functional subnetworks of the brain across 74 healthy individuals. Plots show the AUC of the classifier against the number of dimensions D for feature representations from multi-node2vec, node2vec, DeepWalk, LINE, and the spectral decomposition.

layers. In this way, we add b layers of randomly connected nodes that act as noise against the structure present in the 74 individuals in the study. We set $b = 10$ and 20 and re-ran all of the methods with the same parameter settings as in the original study.

As can be seen in Figures 6 and 7, the competing methods are dramatically affected by the addition of noisy layers; whereas, multi-node2vec is robust to noise. For both $b = 10$ and $b = 20$, all three runs of multi-node2vec outperforms competing methods for seven out of nine of the classification studies. In particular, multi-node2vec has clear advantages over the competing methods in the *subcortical*, *salience*, *sensory-motor (hand)*, and *fronto-parietal task control* regions. Importantly, multi-node2vec's performance is not strongly affected by the addition of more noisy layers suggesting that the features identified by the method align with the true 74 layers of the population. We find that the LINE method is most affected by noise, followed by node2vec. It would be interesting to investigate why this is the case in future analyses.

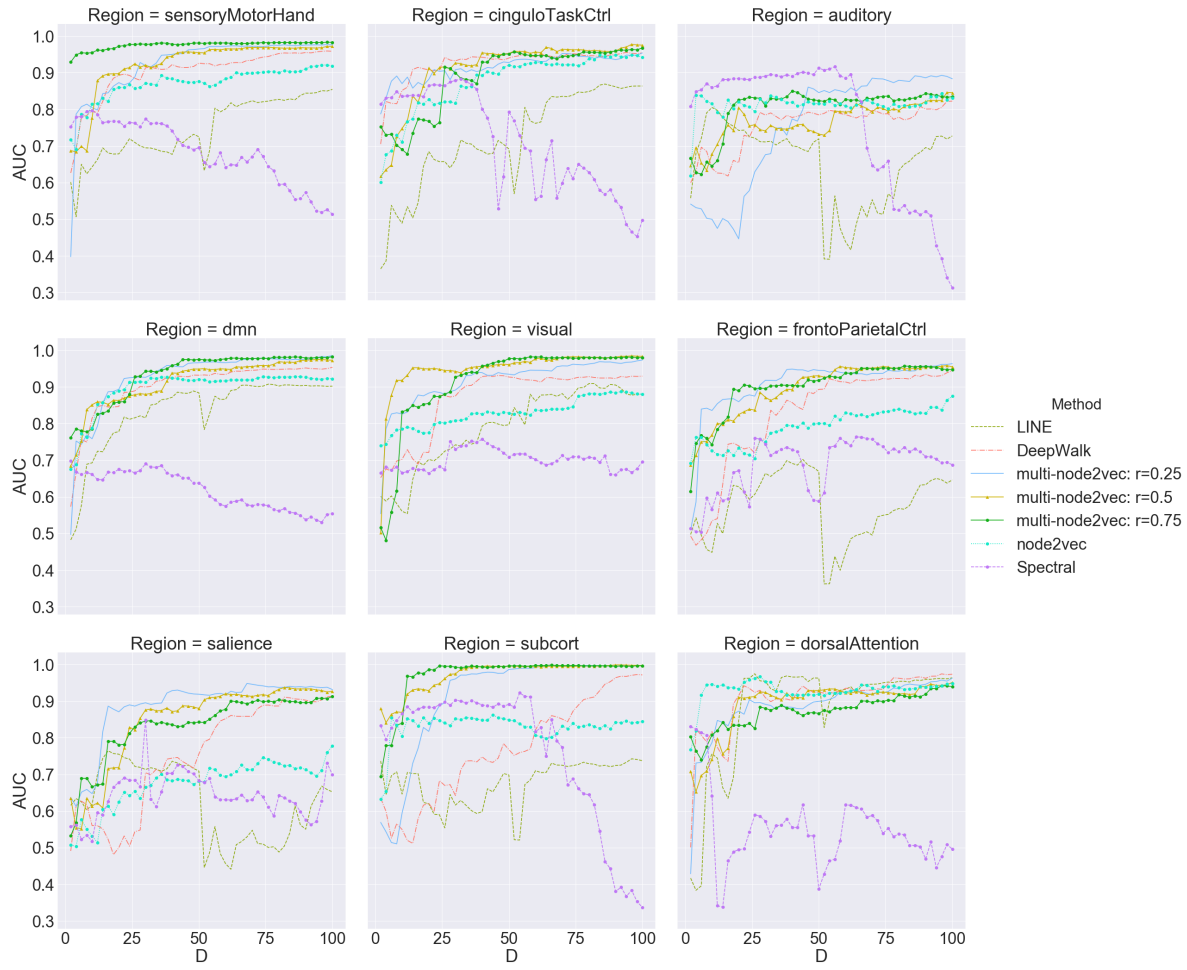

Figure 6: The AUC of a one vs. all logistic regression classifier for the nine major functional subnetworks of the brain across all 74 healthy individuals and 10 layers of noise. Plots show the AUC of the classifier against the number of dimensions D for feature representations from multi-node2vec, node2vec, DeepWalk, LINE and spectral decomposition.

These results, in combination to the clustering results from the previous section, provide strong evidence that the features engineered from multi-node2vec provide biologically relevant information about the functional organization of the brain, and is generally robust to moderate amounts of noisy layers.

7 Simulation Study

We further assess the strengths and weaknesses of multi-node2vec by investigate its performance on a test bed of simulated networks. The focus of our simulation study is threefold: (i) to analyze the features learned by multi-node2vec on well-structured multilayer networks with varying signal and number of layers, (ii) to investigate the effects of the context size parameter on the method, and (iii) to analyze the scalability of multi-node2vec for networks with a large number of nodes and/or layers.

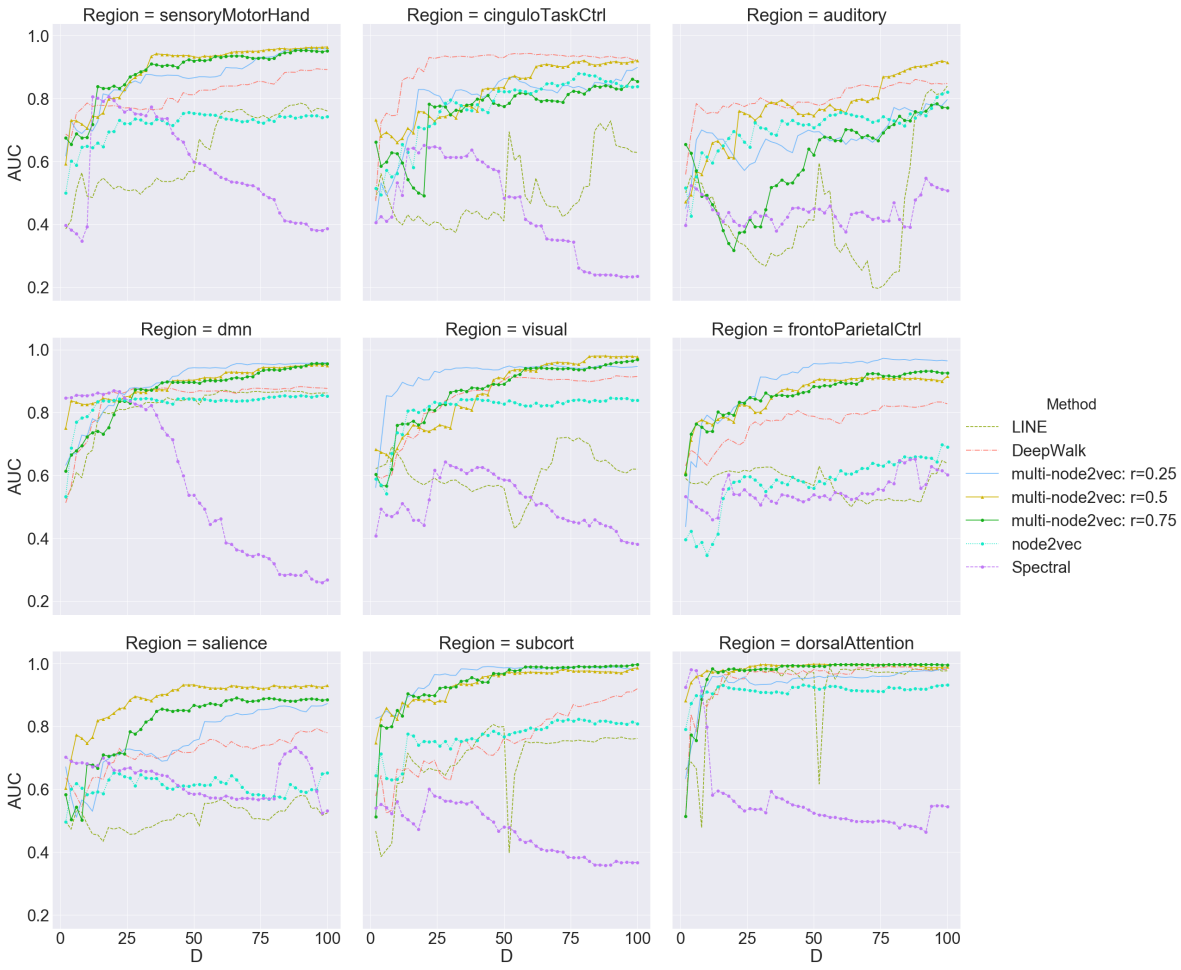

Figure 7: The AUC of a one vs. all logistic regression classifier for the nine major functional subnetworks of the brain all 74 healthy individuals and 20 layers of noise. Plots show the AUC of the classifier against the number of dimensions D for feature representations from multi-node2vec, node2vec, DeepWalk, LINE and spectral decomposition.

Throughout this study we generate unweighted multilayer networks using a multilayer generalization of the planted partition model, described briefly as follows. Each layer of the multilayer network contains $n = 264$ nodes to match the fMRI networks from Section 6. Nodes were placed deterministically into c equally-sized communities. For each layer, edges are placed randomly between two nodes of the same community with probability p_{in} and edges are placed between two nodes of differing communities with probability p_{out}. With this construction, each layer of the generated network has the same community structure across layers. This graph model is a special case of the multilayer stochastic block model (MSBM) considered [25; 53; 61]. The planted partition model is a widely-studied network model where nodes of the same community are stochastically equivalent. For our analysis, nodes of the same community are expected to have similar features with one another and different features than nodes from other communities. This model therefore provides a well-structured multilayer network for which we can study multi-node2vec.

For each of the following studies, we set $p_{in} = 0.49$ to match the average degree of the functional brain networks. To assess the relevance of the features identified by multi-node2vec, we compare the clusters obtained from the k-means algorithm on the feature matrix with the true community labels of the network and calculate the adjusted rand score as a measurement of match between the two partitions. For each simulation, we replicate the study 30 times and report the average adjusted rand score. The results for each simulation is presented in Figure 8 and discussed below.

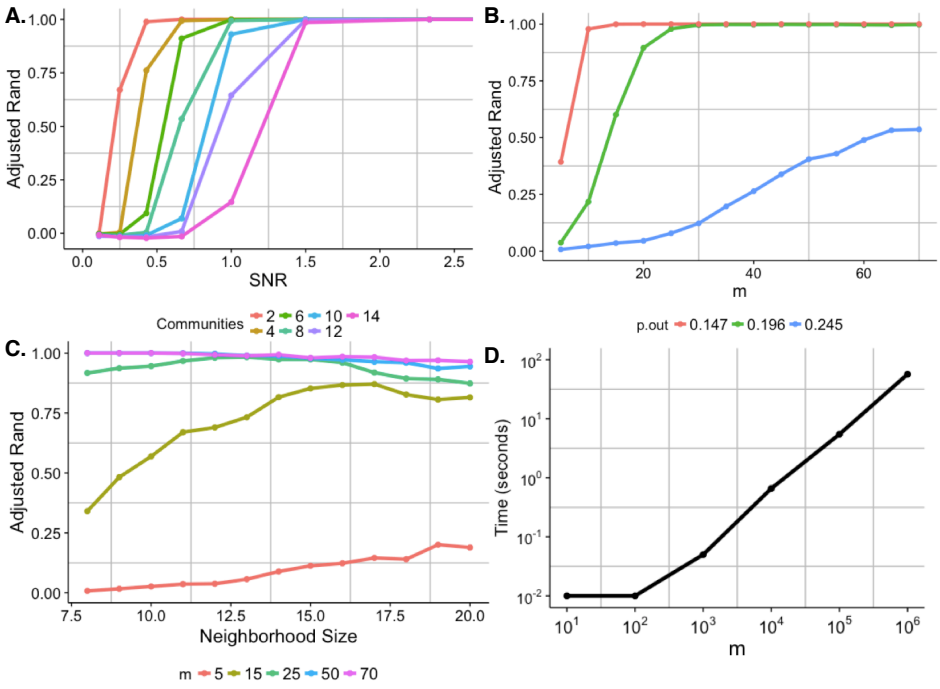

Figure 8: Simulation results from the numerical study described in Section 6. All simulations are repeated 30 times and the average is shown. **A.** The adjusted rand index score of the clusters identified by k-means clustering on the identified feature matrix from multi-node2vec applied to the multilayer stochastic block model as a function of the signal to noise ratio: SNR $= p_{in}/p_{out} - 1$ and **B.** across the number of layers in the network. **C.** The adjusted rand index score of the clusters identified by k-means clustering on the identified feature matrix from multi-node2vec as a function of the neighborhood size input to the algorithm. **D.** The average time (in seconds) required by multi-node2vec on multilayer random graphs with 10 nodes in each layer and m layers. Notably, networks with 1 million layers required just 58 seconds.

7.1 Community Strength

We first investigate the effects of the strength of community structure on multi-node2vec. To do so, we varied the out-group probabilities p_{out} to be between 10% - 90% of p_{in} and assess the performance of the algorithm over values of the signal to noise ratio (SNR) $= p_{in}/p_{out} - 1$. We simulated multilayer networks like this with 74 layers, across $c = 2$ to 14

communities per layer. Results are shown in plot **A.** of Figure 8. We observe that as the disparity between in-group and out-group probabilities increased, the feature embeddings more clearly represented the community structure in the graph. Furthermore, across all values of p_{out} the performance of multi-node2vec improved as the number of communities decreased. For multilayer networks with 2 communities, the feature embeddings perfectly represented the community structure for values of SNR greater than or equal to 0.4. Networks with 14 communities per layer required SNR greater than 2.0 to achieve the same result. These results provide evidence that the feature embeddings identified by multi-node2vec are able to efficiently capture the community structure of multilayer networks.

7.2 Effect of the Number of Layers

We next analyze the effect of the number of layers on the multi-node2vec algorithm. In this simulation, we generated multilayer graphs from the planted partition model with $m =$ 5, 10, 15, 65, 74. As before, we fixed $p_{in} = 0.49$ and varied $p_{out} = 0.245, 0.196$, and 0.147 to match the best three values from the community strength simulations. We report the average adjusted rand from 30 replications on networks with $c = 12$ communities in plot **B.** of Figure 8. For all three values of p_{out}, the performance of multi-node2vec consistently improves across an increasing number layers. This result supports the belief that each layer provides additional neighborhood information for each node from which the multi-node2vec algorithm can efficiently learn.

7.3 Sensitivity to Context Size

To test the effect of neighborhood size, we ran simulations of the planted partition model multilayer networks with $m = 5$, 15, 25, 50, and 74 over a range of 8 - 20 nodes per neighborhood with $p_{out} = 0.245$. We plot the average adjusted rand of the clusters identified on the feature matrix for networks with $c = 12$ communities in the plot **C.** of Figure 8. We find that the algorithm improves with an increasing context size; however, the number of layers in the network has more impact on the performance of the algorithm. Indeed, when $m \geqslant 25$, the neighborhood size does not significantly affect (if at all) the performance of the algorithm. On the other hand, for a small number of layers (say, $m = 5$) the increasing the context size plays a more important role in its identified features. Thus, for multilayer networks with a large enough of layers, the context size will not dramatically affect the results of multi-node2vec, but in networks with fewer than 25 layers, one should carefully tune this parameter.

7.4 Scalability

Identifying a neighborhood for the bag of nodes needed for the algorithm relies upon a random walk strategy, which can be done in constant time using alias sampling (as done in the node2vec algorithm). The optimization part of the algorithm turns out to be linear in the number of distinct nodes in the multilayer network. Notably, this is drastically faster than the spectral decomposition of the network, which in the best case scenario is of cubic in the unique number of nodes. To show this empirically, we consider multilayer networks

with $n = 10$ unique nodes in each of m total layers. We apply multi-node2vec on planted partition networks across a range the number of layers m from 10 to 1 million layers. We calculate the amount of time (in seconds) required for multi-node2vec with fixed $k = D = 5$ on 30 replications and report the average time in the plot **D.** of Figure 8. For networks with 1 million layers, multi-node2vec took on average only 58 seconds. We note that the complexity of multi-node2vec as a function of n is also linear, and this is justified with the scalability analysis in [24]. This figure suggests that the multi-node2vec algorithm is linear in the number of layers in the network, and provides evidence that this algorithm is well-suited for embedding massive multilayer networks.

8 Discussion

In this paper, we introduced the multi-node2vec algorithm, a fast network embedding technique for complex multilayer networks. Our numerical studies illustrated the speed and efficacy of the algorithm on large multilayer networks. This work motivates several areas of future work. For example, an important next step is to incorporate partial supervision for the detection of relevant features that depend on the application under investigation. Recent work like Kipf and Welling [30] for semi-supervised feature engineering on static networks may provide a principled first step in the investigation for multilayer networks. We furthermore believe that it will be fruitful to thoroughly compare and contrast feature engineering methods like multi-node2vec with the results of multilayer community detection methods so as to better understand the discovered features. Furthermore, though not explicitly considered here, multi-node2vec is readily applicable to dynamic networks, an example of multilayer networks where the ordering of layers depends on time. This work will require incorporating appropriate notions of conditional dependence between the layers that replace the conditional independence assumptions applied here. Finally, our theoretical analysis of the multi-node2vec algorithm motivates further work in understanding the relationship between algorithms based on neural networks and deep learning with more traditional machine learning tasks such as matrix factorization. We believe that more work should be done in this area to fully understand the theoretical underpinnings of deep learning and neural networks.

The multi-node2vec technique further has potential for ground-breaking discovery in the field of network neuroscience. By specifying a multilevel framework that (i) models weighted networks, (ii) does not require temporal ordering of the layers, and (iii) is robust to noisy layers, multi-node2vec enables the study of networks that vary across individuals and cognitive tasks. Our proposed algorithm provides a method that potentially learns significant neurological variation among brains, which we hope will be further the investigation of individual differences and disease.

9 Appendix

Proof of Theorem 2. Since multi-node2vec, node2vec, and DeepWalk all use Skip-Gram on identified neighborhoods, it will suffice to show that the transition probabilities of the random walks used to identify the neighborhoods for each method are equal under the stated

conditions to prove Theorem 2. We begin by proving part (a) for general $p, q, r > 0$. Let $\pi_{u,v}$ denote the unnormalized transition probability of the random walk traveling from $u \to v$ based on the application of node2vec on the graph $\widetilde{\mathbf{G}}_N(r)$. Similarly let $\pi^*_{u,v}$ denote this unnormalized transition probability of the random walk based on the application of multi-node2vec to $\mathbf{G}^m_{\mathcal{N}}$. Then by the law of total probability we have

$$\begin{aligned}\pi^*_{u,v} &:= Z \cdot P_{\mathcal{G}^m_{\mathcal{N}}}(u_{j+1} = u \mid u_j = v) = Z \cdot \sum_{\ell} \sum_{\ell'} w_{\ell,\ell'}(v, x) P(\ell_{i-1} = \ell) \\ &= \beta_{pq}(t, v) \sum_{\ell} w_{\ell}(u, v) + m(m_u - 1) r^{-1} C_u \mathbb{I}(v = u)\end{aligned}$$

Note that $\beta_{pq}(t, v) = 1$ when $v = u$. It follows that $\pi^*_{u,v} = \pi_{u,v}$ and thus part (a) is proved. Part (b) is proven in an analogous fashion by taking $\pi_{u,v}$ as the transition probability for the random walk associated with DeepWalk on the graph $\widetilde{\mathbf{G}}_N(r)$ and noting that $\beta_{pq}(t, v) \equiv 1$ when $p = q = 1$. ■

Proof of Theorem 3. By applying the result of Theorem 2, we can directly apply Theorems 2.1 - 2.3 and result (8) from [47] directly to prove the desired results of the theorem. ■

References

[1] Achard, S., R. Salvador, B. Whitcher, J. Suckling, and E. Bullmore (2006). A resilient, low-frequency, small-world human brain functional network with highly connected association cortical hubs. *The Journal of neuroscience 26*(1), 63–72.

[2] Bassett, D. S. and E. Bullmore (2006). Small-world brain networks. *The neuroscientist 12*(6), 512–523.

[3] Bassett, D. S., A. Meyer-Lindenberg, S. Achard, T. Duke, and E. Bullmore (2006). Adaptive reconfiguration of fractal small-world human brain functional networks. *Proceedings of the National Academy of Sciences 103*(51), 19518–19523.

[4] Bassett, D. S., N. F. Wymbs, M. A. Porter, P. J. Mucha, J. M. Carlson, and S. T. Grafton (2011). Dynamic reconfiguration of human brain networks during learning. *Proceedings of the National Academy of Sciences 108*(18), 7641–7646.

[5] Bassett, D. S., M. Yang, N. F. Wymbs, and S. T. Grafton (2015). Learning-induced autonomy of sensorimotor systems. *Nature neuroscience 18*(5), 744–751.

[6] Betzel, R. F. and D. S. Bassett (2016). Multi-scale brain networks. *NeuroImage*.

[7] Biswal, B. B., M. Mennes, X.-N. Zuo, S. Gohel, C. Kelly, S. M. Smith, C. F. Beckmann, J. S. Adelstein, R. L. Buckner, S. Colcombe, et al. (2010). Toward discovery science of human brain function. *Proceedings of the National Academy of Sciences 107*(10), 4734–4739.

[8] Boccaletti, S., G. Bianconi, R. Criado, C. Del Genio, J. Gómez-Gardeñes, M. Romance, I. Sendina-Nadal, Z. Wang, and M. Zanin (2014). The structure and dynamics of multilayer networks. *Physics Reports 544*(1), 1–122.

[9] Braun, U., A. Schäfer, H. Walter, S. Erk, N. Romanczuk-Seiferth, L. Haddad, J. I. Schweiger, O. Grimm, A. Heinz, H. Tost, et al. (2015). Dynamic reconfiguration of frontal brain networks during executive cognition in humans. *Proceedings of the National Academy of Sciences 112*(37), 11678–11683.

[10] Bressler, S. L. and V. Menon (2010). Large-scale brain networks in cognition: emerging methods and principles. *Trends in cognitive sciences 14*(6), 277–290.

[11] Bühlmann, P. and S. Van De Geer (2011). *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media.

[12] Bullmore, E. and O. Sporns (2009). Complex brain networks: graph theoretical analysis of structural and functional systems. *Nature Reviews Neuroscience 10*(3), 186–198.

[13] Bullmore, E. and O. Sporns (2012). The economy of brain network organization. *Nature Reviews Neuroscience 13*(5), 336–349.

[14] Cardillo, A., J. Gómez-Gardeñes, M. Zanin, M. Romance, D. Papo, F. Del Pozo, and S. Boccaletti (2013). Emergence of network features from multiplexity. *Scientific reports 3*.

[15] Chai, X. J., A. N. Castañón, D. Öngür, and S. Whitfield-Gabrieli (2012). Anticorrelations in resting state networks without global signal regression. *Neuroimage 59*(2), 1420–1428.

[16] Cole, M. W., T. Yarkoni, G. Repovš, A. Anticevic, and T. S. Braver (2012). Global connectivity of prefrontal cortex predicts cognitive control and intelligence. *The Journal of Neuroscience 32*(26), 8988–8999.

[17] Daianu, M., E. L. Dennis, N. Jahanshad, T. M. Nir, A. W. Toga, C. R. Jack, M. W. Weiner, and P. M. Thompson (2013). Alzheimer's disease disrupts rich club organization in brain connectivity networks. In *2013 IEEE 10th International Symposium on Biomedical Imaging*, pp. 266–269. IEEE.

[18] De Domenico, M., A. Lancichinetti, A. Arenas, and M. Rosvall (2015). Identifying modular flows on multilayer networks reveals highly overlapping organization in interconnected systems. *Physical Review X 5*(1), 011027.

[19] Ferriani, S., F. Fonti, and R. Corrado (2013). The social and economic bases of network multiplexity: Exploring the emergence of multiplex ties. *Strategic Organization 11*(1), 7–34.

[20] Fornito, A., A. Zalesky, D. S. Bassett, D. Meunier, I. Ellison-Wright, M. Yücel, S. J. Wood, K. Shaw, J. O'Connor, D. Nertney, et al. (2011). Genetic influences on cost-efficient organization of human cortical functional networks. *The Journal of Neuroscience 31*(9), 3261–3270.

[21] Gallagher, B. and T. Eliassi-Rad (2010). Leveraging label-independent features for classification in sparsely labeled networks: An empirical study. In *Advances in Social Network Mining and Analysis*, pp. 1–19. Springer.

[22] Garrity, A. G., G. D. Pearlson, K. McKiernan, D. Lloyd, K. A. Kiehl, and V. D. Calhoun (2007). Aberrant default mode functional connectivity in schizophrenia. *American journal of psychiatry*.

[23] Goyal, P. and E. Ferrara (2018). Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems 151*, 78–94.

[24] Grover, A. and J. Leskovec (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864. ACM.

[25] Han, Q., K. Xu, and E. Airoldi (2015). Consistent estimation of dynamic and multi-layer block models. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1511–1520.

[26] He, Y., Z. J. Chen, and A. C. Evans (2007). Small-world anatomical networks in the human brain revealed by cortical thickness from mri. *Cerebral cortex 17*(10), 2407–2419.

[27] Henderson, K., B. Gallagher, L. Li, L. Akoglu, T. Eliassi-Rad, H. Tong, and C. Faloutsos (2011). It's who you know: graph mining using recursive structural features. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 663–671. ACM.

[28] Hoff, P. D., A. E. Raftery, and M. S. Handcock (2002). Latent space approaches to social network analysis. *Journal of the american Statistical association 97*(460), 1090–1098.

[29] Kinnison, J., S. Padmala, J.-M. Choi, and L. Pessoa (2012). Network analysis reveals increased integration during emotional and motivational processing. *The Journal of Neuroscience 32*(24), 8361–8372.

[30] Kipf, T. N. and M. Welling (2016). Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

[31] Kivelä, M., A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter (2014). Multilayer networks. *Journal of Complex Networks 2*(3), 203–271.

[32] Lambiotte, R., J.-C. Delvenne, and M. Barahona (2008). Laplacian dynamics and multiscale modular structure in networks. *arXiv preprint arXiv:0812.1770*.

[33] Lee, J. D., M. Simchowitz, M. I. Jordan, and B. Recht (2016). Gradient descent only converges to minimizers. In *Conference on Learning Theory*, pp. 1246–1257.

[34] Levy, O. and Y. Goldberg (2014). Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pp. 2177–2185.

[35] Lynall, M.-E., D. S. Bassett, R. Kerwin, P. J. McKenna, M. Kitzbichler, U. Muller, and E. Bullmore (2010). Functional connectivity and brain networks in schizophrenia. *The Journal of Neuroscience 30*(28), 9477–9487.

[36] Mayer, A. R., M. V. Mannell, J. Ling, C. Gasparovic, and R. A. Yeo (2011). Functional connectivity in mild traumatic brain injury. *Human brain mapping 32*(11), 1825–1835.

[37] Mayer, A. R., D. Ruhl, F. Merideth, J. Ling, F. M. Hanlon, J. Bustillo, and J. Cañive (2013). Functional imaging of the hemodynamic sensory gating response in schizophrenia. *Human brain mapping 34*(9), 2302–2312.

[38] McPherson, M., L. Smith-Lovin, and J. M. Cook (2001). Birds of a feather: Homophily in social networks. *Annual review of sociology 27*(1), 415–444.

[39] Medaglia, J. D., M.-E. Lynall, and D. S. Bassett (2015). Cognitive network neuroscience. *Journal of cognitive neuroscience*.

[40] Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[41] Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado, and J. Dean (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119.

[42] Mucha, P. J., T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela (2010). Community structure in time-dependent, multiscale, and multiplex networks. *Science 328*(5980), 876–878.

[43] Muldoon, S. F. and D. S. Bassett (2016). Network and multilayer network approaches to understanding human brain dynamics. *Philosophy of Science 83*(5), 710–720.

[44] Pennington, J., R. Socher, and C. D. Manning (2014). Glove: Global vectors for word representation. In *EMNLP*, Volume 14, pp. 1532–1543.

[45] Perozzi, B., R. Al-Rfou, and S. Skiena (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 701–710. ACM.

[46] Power, J. D., A. L. Cohen, S. M. Nelson, G. S. Wig, K. A. Barnes, J. A. Church, A. C. Vogel, T. O. Laumann, F. M. Miezin, B. L. Schlaggar, et al. (2011). Functional network organization of the human brain. *Neuron 72*(4), 665–678.

[47] Qiu, J., Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang (2018). Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pp. 459–467. ACM.

[48] Rubinov, M. and O. Sporns (2010). Complex network measures of brain connectivity: uses and interpretations. *Neuroimage 52*(3), 1059–1069.

[49] Smith, S. M., K. L. Miller, G. Salimi-Khorshidi, M. Webster, C. F. Beckmann, T. E. Nichols, J. D. Ramsey, and M. W. Woolrich (2011). Network modelling methods for fmri. *Neuroimage 54*(2), 875–891.

[50] Sporns, O. (2011). *Networks of the Brain*. MIT press.

[51] Sporns, O. (2014). Contributions and challenges for network models in cognitive neuroscience. *Nature neuroscience 17*(5), 652–660.

[52] Sporns, O. and R. F. Betzel (2016). Modular brain networks. *Annual review of psychology 67*, 613.

[53] Stanley, N., S. Shai, D. Taylor, and P. Mucha (2016). Clustering network layers with the strata multilayer stochastic block model. *IEEE*.

[54] Stillman, P. E., J. D. Wilson, M. J. Denny, B. A. Desmarais, S. Bhamidi, S. J. Cranmer, and Z.-L. Lu (2017). Statistical modeling of the default mode brain network reveals a segregated highway structure. *Scientific reports 7*(1), 11694.

[55] Strano, E., S. Shai, S. Dobson, and M. Barthelemy (2015). Multiplex networks in metropolitan areas: generic features and local effects. *Journal of The Royal Society Interface 12*(111), 20150651.

[56] Tang, J., M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei (2015). Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, pp. 1067–1077. International World Wide Web Conferences Steering Committee.

[57] van den Heuvel, M. P., R. S. Kahn, J. Goñi, and O. Sporns (2012). High-cost, high-capacity backbone for global brain communication. *Proceedings of the National Academy of Sciences 109*(28), 11372–11377.

[58] van den Heuvel, M. P. and O. Sporns (2011). Rich-club organization of the human connectome. *The Journal of neuroscience 31*(44), 15775–15786.

[59] van den Heuvel, M. P., O. Sporns, G. Collin, T. Scheewe, R. C. Mandl, W. Cahn, J. Goñi, H. E. H. Pol, and R. S. Kahn (2013). Abnormal rich club organization and functional brain dynamics in schizophrenia. *JAMA psychiatry 70*(8), 783–792.

[60] Wilson, J. D., M. J. Denny, S. Bhamidi, S. J. Cranmer, and B. A. Desmarais (2017). Stochastic weighted graphs: Flexible model specification and simulation. *Social Networks 49*, 37–47.

[61] Wilson, J. D., J. Palowitch, S. Bhamidi, and A. B. Nobel (2017). Community extraction in multilayer networks with heterogeneous community structure. *The Journal of Machine Learning Research 18*(1), 5458–5506.

[62] Zhang, K., B. Johnson, M. Gay, S. G. Horovitz, M. Hallett, W. Sebastianelli, and S. Slobounov (2012). Default mode network in concussed individuals in response to the ymca physical stress test. *Journal of neurotrauma 29*(5), 756–765.

[63] Zhou, Y., M. P. Milham, Y. W. Lui, L. Miles, J. Reaume, D. K. Sodickson, R. I. Grossman, and Y. Ge (2012). Default-mode network disruption in mild traumatic brain injury. *Radiology 265*(3), 882–892.