Disease Protein Prediction with Graph Convolutional Networks

E. Sabri Eyuboglu and Pierce B. Freeman

November 2017

1 Introduction

Human phenotypes – i.e. our characteristics, conditions and diseases – are not merely a product of our genetic constitution, but rather arise out of an intricate system of interactions between the proteins and other molecules in our cells. [1] This fact has inspired a huge effort to document and understand the network of those protein-protein interactions which we'll refer to collectively as the protein-protein interaction network or the human interactome. The protein-protein interaction network can be intuitively represented as an undirected graph: the nodes are proteins and each edge represents a binary physical interaction between proteins.

Like other human phenotypes, the mechanics behind human diseases can often be traced back to complex pathways of interacting proteins in our cells – pathways that are encoded in the human interactome. This is to say, a disease phenotype is not so much a product of defect in a single gene or protein, as much as it is a consequence of mutations affecting a complex system interacting proteins in human cells.[2] This set of proteins active in the manifestation of a disease is collectively referred to as a disease pathway. Identifying proteins active in a disease pathway can deepen our understanding of the disease and facilitate drug and treatment discovery. The problem of **Disease Protein Prediction** (DPP), that of predicting which proteins may be active in a disease pathway, is, thus, one of great interest. As we discuss in section 3, because diseases are caused by alterations in *interacting* proteins with human cells, disease pathways exhibit connectivity patterns in the protein-protein interaction network. As a result, most recent methods of computational disease protein prediction have used the protein-protein interaction network as a primary dataset.

In this work, we use graph-based neural networks to pose a solution to disease protein prediction. Graph Convolutional Networks (GCN) recently proposed by Kipf & Welling and Defferrard *et al.* present a scalable approach to semi-supervised classification of graph nodes.[3,4] As we show in section 2, the disease protein prediction problem can be framed quite naturally as a semi-supervised classification problem on the protein-protein interaction network and, thus, GCNs can be naturally applied to disease protein prediction. Furthermore, as we discuss in sections 3, 4 and 5 the power of GCNs to uncover latent patterns of both neighborhood and higher-order connectivity make them particularly well-suited for disease protein prediction. Empirically, we find that GCNs perform remarkably well on the disease protein prediction problem: our results show Graph Convolutional Networks decisively outperform existing representation learning methods that use neighborhood preserving network-embeddings (node2vec[5]). The application of Graph Convolutional Networks is, thus, a very promising area for continued work in disease protein prediction.

2 Semi-Supervised Node Classification

The disease-protein prediction problem can be formulated naturally as a semi-supervised node classification problem. In this section we provide a formal problem formulation.

In a semi-supervised classification problem, our training dataset consists of both labeled and unlabeled data. We have a training set consisting of m examples $x^{(i)} \in \mathbb{R}^n$ where n is the number of features:

$$\{x^{(1)}, x^{(2)} ... x^{(m)}\} = X \in \mathbb{R}^{m \times n} \tag{1}$$

In addition, we have a set of m label variables $y^{(i)} \in [0, k-1] \cup \{\text{NA}\}$ where k is the number of classes in our classification problem and NA indicates that that $x^{(i)}$ example is unlabeled:

$$\{y^{(1)}, y^{(2)} ... y^{(m)}\} = Y \in ([0, k-1] \cup \{\text{NA}\})^m \tag{2}$$

The goal of semi-supervised classification is to learn some function $f(X) = \hat{Y} \in [0, k-1]^m$ using both labeled and unlabeled examples as training that minimizes some semi-supervised loss function $\mathcal{L}(Y, \hat{Y})$ that considers only labeled points.[6]

Semi-supervised node classification is simple extension of the vanilla semi-supervised learning problem. Let's assume we have partially labeled graph $G = (V, E)$ defined by some adjacency matrix $A \in \mathbb{R}^{m \times m}$. In this setting, each node V_i

[1] Barabasi, Albert-Laszlo, and Zoltan N. Oltvai. "Network biology: understanding the cell's functional organization." Nature reviews genetics 5.2 (2004): 101-113.

[2] Li Y, Agarwal P. A Pathway-Based View of Human Diseases and Disease Relationships. Hide W, ed. PLoS ONE. 2009;4(2):e4346. doi:10.1371/journal.pone.0004346.

[3] Thomas N. Kipf and (2016). Semi-Supervised Classification with Graph Convolutional Networks. CoRR, abs/1609.02907

[4] Michael Defferrard and (2016).Convolutional Neural Networks on Graphs with Fast Localized Spectral. CoRR, abs/1606.09375

[5] Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2016.

[6] Zhu & Goldberg; Introduction to Semi-Supervised Learning. 2009. MIT Press.

has an associated feature vector $x^{(i)}$, and if the node is labeled an associated label vector $y^{(i)}$. The feature vector $x^{(i)}$ should be derived from characteristics of the node relevant to the learning problem. For example, if our aim was to find communities in some social network G, a relevant feature to include in the feature vector $x^{(i)}$ would be the age of person associated with a node. Thus the goal here is to learn some function $f(X, A) = \hat{Y} \in [0, k-1]^m$ that minimizes some loss function $\mathcal{L}(Y, \hat{Y})$.

Disease protein prediction is semi-supervised node classification on the protein-protein interaction network. Our model will consist of two inputs: an adjacency matrix describing the protein-protein interaction network and a feature matrix set to the identity. As output it will produce a single binary label vector indicating whether or not each protein is involved in the disease in question.

Adjacency matrix Let's call $P = (V, E)$ our protein-protein interaction network. In this formulation V is the set of all proteins in the human interactome and E is the set of all bijective interactions in the interactome. Our network P could be fully described by an adjacency matrix $A \in \mathbb{R}^{m \times m}$ where $m = |V| \approx 21,500$ (the number of proteins in the human interactome).

Feature Matrix Since the proteins in question don't have rigorously established features, we use the identity matrix to nullify the effect of a feature matrix.

$$\{x^{(1)}, x^{(2)}...x^{(m)}\} = I = X \in \mathbb{R}^{m \times n} \tag{3}$$

Label Vector Let's say we have a particular disease of interest and an associated disease pathway D consisting of proteins known to be associated with the disease (Note: D represents the set of disease proteins known today, not the absolute, true set of disease proteins, which we can denote $\hat{D}$). For each of those proteins $i \in D$ we can definitively assign a label $y^{(i)} = 1$. But what about every other protein $j \notin D$? One of the fundamental challenges in Disease Protein Prediction arises from the fact that our input disease pathways are incomplete ($D \subset \hat{D}$). As a result, we can't simply assign $y^{(j)} = 0$ for all $j \notin D$. All $x^{(j)}$ for $j \notin D$ are hence unlabeled.

$$y^{(i)} = \begin{cases} 1 & \text{if } i \in d \\ NA & \text{if } i \notin d \end{cases} \tag{4}$$

It should now be clear how disease protein prediction constitutes semi-supervised binary node classification.

One issue does remain however. Under this formulation we have a small set of positively labeled nodes, a huge set of unlabeled nodes, but zero negatively labeled nodes. To address the lack of negative labels, we propose a simple approach: choose k random unlabeled nodes and label them as negative. At first glance, this may seem misguided at best or catastrophic at worst — if we follow this strategy a large majority of our labels will be randomly chosen. However, let's consider the probability that a protein j chosen at uniformly at random from $|V|$ is active in the true disease pathway $\hat{D}$:

$$P(j \in D) = \frac{|\hat{D}|}{|V|} \tag{5}$$

We know that $|V| \approx 21,500$ and can estimate $|\hat{D}|$ with the maximum size of D over all our dataset of diseases. Using this approximation we get: $P(j \in D) \approx 0.2\%$. With such a low probability of error, we can view our random negative labels as accurate labels with some reasonable amount of random noise. Indeed, most supervised learning problems involve some noise in the labels and robust learning algorithms should be able to handle that noise. (Note: We leave the size of k as a tuneable hyper-parameter that is explored in more depth.)

Our goal is thus to learn some function $f(X, P) = \hat{Y} \in [0, 1]^m$ that minimizes some loss function $\mathcal{L}(Y, \hat{Y})$. What differentiates our approach from vanilla semi-supervised learning methods is that it also takes as input and utilizes a graph as a training data source.

3 Disease Pathway Connectivity and the PPI Network

As we mentioned in the introduction, most human diseases arise from a set of active disease proteins that have either been mutated directly, or whose interactions have been affected by mutations. As a result, the PPI interaction network is a dataset of great interest in the protein-prediction problem. In this section, we briefly outline the network properties of the PPI network and offer a more in depth coverage of connectivity patterns of disease pathways in the protein-protein interaction network.

3.1 Protein-Protein Interaction Network Properties

The protein-protein interaction network is an undirected graph where each node represents a protein and each edge records a known binary physical interaction. Within this model, there are $|V| = 21,557$ nodes with $|E| = 342,353$ edges. Our PPI network is, thus, extremely sparse with a density of $D = 0.00147$.

Like many real-world networks, cellular interaction networks tend to exhibit a heavy-tailed, power-law degree distribution – that is, they exhibit high variability in node degree, and have a significant number of nodes of very high degree.[7] To better understand the PPI network we are using as our primary data-source, we computed our PPI network's degree distribution and assessed its fit to the power-law. As we can see in Figure 1 below, our PPI network roughly follows a power-law distribution with $\alpha = 2.04$. We estimated this α value using maximum-likelihood estimation.

$$\ell(\alpha; x_1, ..., x_n) = \sum_{i=1}^{n} \ln(P(X = x_i|\alpha)) \tag{6}$$

[7] Barabasi, Albert-Laszlo, and Zoltan N. Oltvai. "Network biology: understanding the cell's functional organization." Nature reviews genetics 5.2 (2004): 101-113. APA

where x_i is the degree count for the i^{th} node in the network.

$$P(X = x_i) = \frac{\alpha - 1}{x_{min}} \left(\frac{x_i}{x_{min}} \right)^{-\alpha}$$

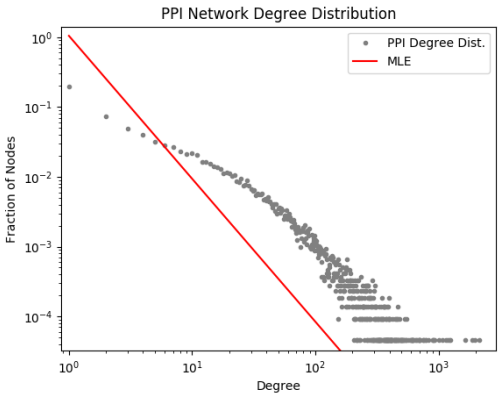


Figure 1: Our protein-protein interaction network approximately follows a power-law distribution with predicted $\alpha = 2.04$.

That our PPI-network's degree distribution roughly follows the power-law suggests the existence of hub proteins that tend to interact with many, many proteins in the cell: the protein with maximal degree interacts with 10% of the proteins in the network. It also implies that a very large fraction of proteins have very few documented interactions. Indeed, roughly 20% of proteins have only one documented interaction, and 50% have less than 10. These degree distributions are important to keep in mind as we consider the potential effectiveness of our disease protein prediction methods.

Our protein-protein interaction network also exhibits the small-world phenomenon – the approximate diameter of the network is 8, computed by running breadth-first search from 1000 test nodes. [8] Furthermore, we also computed a relatively high average clustering coefficient at $C = 0.207$. This average clustering coefficient is substantially higher than that of random preferential attachment networks with the same number of nodes ($C = 0.009$) and random networks with the PPI network's degree distribution ($C = 0.005$). Finally, our PPI network is characterized by one, very large connected component which includes 99.8% of the proteins in the network.

It is important to understand these fundamental properties of our PPI network (i.e. degree distribution, clustering coefficient, diameter, etc.) prior to designing and implementing disease protein prediction models that use the PPI network as a primary dataset. Also important is understanding how known disease proteins fit into the PPI network we just described. In the following two subsections, we report important metrics of connectivity in disease pathways and discuss the existence of higher-order connectivity patterns of disease patwhays in the PPI network.

[8] Vazquez, Alexei. "Protein interaction networks." Neuroproteomics. Boca Raton, FL (2010).

3.2 Local-Neighborhood Connectivity of Disease Pathways in PPI Network

Until recently, it had been widely expected that disease proteins occupy well-defined neighborhoods of the protein-protein interaction network. Indeed, many studies adopted the disease module hypothesis, which suggests that disease pathways compose highly-connected neighborhoods or *disease modules*. For example, in the landmark 2011 Barabási *et al.* review of network-based approaches to human disease, it was suggested that "if we identify a few disease components , the other disease-related components will likely be in their network-based vicinity." [9] The disease module hypothesis found empirical support in the work of Goh *et al.*, who in their paper The Human Disease Network, showed that the number of interactions between proteins of the same disease pathway was ten times higher than random expectation. [10] This finding has been corroborated and extended by the works of others: Ghandi T. *et al.* found that disease causing genes have propensity to be connected in the PPI network and Xu and Li showed that proteins in different diseases with similar phenotypes are prone to interaction. [11] These studies and the disease module hypothesis have inspired a series of disease protein prediction methods that either directly or indirectly search for proteins in the network neighborhood of known disease proteins.[12] (We discuss some of these methods in further depth in section 4.)

In line with the observations of Goh *et al.*, we find that disease pathways are better connected internally than the PPI network at large: disease pathways have a median density of $D = 0.07$ compared to a density of $D = 0.00147$ for the PPI network. Furthermore, disease-pathways have a median conductance of 0.96 in the PPI network, meaning that roughly half of the edges with at least one end-point within the pathway have both end-points in the pathway.

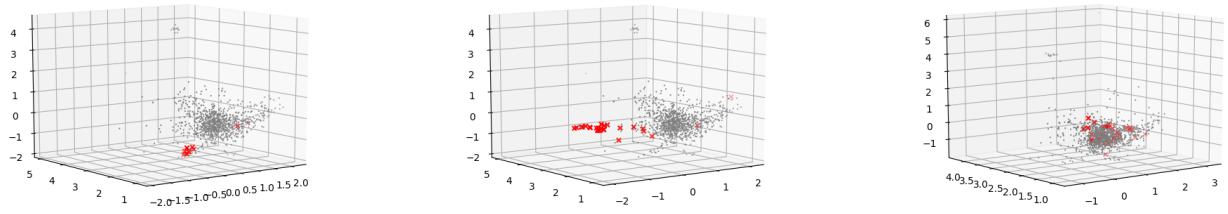
However, recent analysis of the PPI network by Agrawal *et al.* indicates that the disease module hypothesis is hardly a sure bet. Rather, while disease pathways do show somewhat high internal edge connectivity on average – by other network metrics, most disease pathways lack the characteristics of tightly-knit neighborhoods. For example, consider the induced subgraph consisting of all proteins in a disease pathway. When analyzing 519 disease pathways, Agrawal *et al.* found that disease pathway induced subgraphs are highly fragmented: the median number of connected components in a disease pathway induced subgraph is 16, and a median of close to 80% of disease proteins exist

[9] Barabási, Albert-László, Natali Gulbahce, and Joseph Loscalzo. "Network medicine: a network-based approach to human disease." Nature Reviews Genetics 12.1 (2011): 56-68. APA

[10] Goh, Kwang-Il, et al. "The human disease network." Proceedings of the National Academy of Sciences 104.21 (2007): 8685-8690.

[11] Gandhi, T. K. B., et al. "Analysis of the human protein interactome and comparison with yeast, worm and fly interaction datasets." Nature genetics 38.3 (2006): 285-293.

[12] Saket Navlakha, Carl Kingsford; The power of protein interaction networks for associating genes with diseases, Bioinformatics, Volume 26, Issue 8, 15 April 2010, Pages 1057–1063, https://doi.org/10.1093/bioinformatics/btq076



(a) Diamond-Blackfan Anemia (b) Mitochondrial Complex I Deficiency (c) Celiac Disease

Figure 2: 3-dimensional neighborhood preserving embeddings of the PPI network. Proteins corresponding to each disease pathway are plotted in red, and random subset of 1,000 PPI network proteins are plotted in grey. Note: we ran node2vec with $p = q = 1$.

outside of the induced subgraph's largest connected component. [13] The highly-fragmented nature of disease pathways in the PPI network directly challenges the existence of disease modules in the PPI network. Furthermore, while a few disease pathways exhibit exceptionally high spatial localization, 92% of diseases show no significant spatial localization suggesting that most pathways occupy multiple neighborhoods in the PPI network. Below we discuss an approach we implemented for visualizing the neighborhood localization of disease pathways. Our visualizations are in accordance with Agrawal *et al.*'s observations of spatial localization of pathways. Specifically, while a tiny fraction of disease pathways show clear neighborhood localization similar to Diamond-Blackfan Anemia in Figure 2a and Mitochondrial Complex I Deficiency in Figure 2b, the vast majority of diseases resemble Celiac Disease, which shows no obvious neighborhood localization.

In Figure 2, we visualize neighborhood localization of disease pathways using Grover *et al.*'s a method for learning a low dimensional mapping of nodes that preserves network neighborhoods. Specifically, we learn a 3-dimensional embedding of proteins in the PPI network, and plot disease proteins alongside a random subset of proteins from the network. Grover *et als.* approach is very-well suited for producing visualizations of neighborhood localization, because their embedding aims preserves network neighborhoods.[14] Specifically, Grover *et al.*'s feature learning algorithm maximizes an objective function that models the likelihood that the graphs networks are observed given the feature embedding. For a given graph $G = (V, E)$ and dimensionality d we can define an embedding $f \in V \to \mathbb{R}^d$ with parameters $\theta = \mathbb{R}^{|V| \times d}$. Grover *et al.*'s objective function can be written as the following log-conditional likelihood:

$$\sum_{u \in V} \log P(N_s(u)|\theta) \tag{7}$$

where $N_s(u)$ is the neighborhood of u. If we assume (1) that the probabilities that different nodes $v_i, v_j \in V$ belong to the same neighborhood $N_s(u)$ are conditionally independent, and (2) that the probability that a node $v_i \in V$ belongs to a neighborhood $N_s(u)$ is given by the softmax function $\sigma(\theta \cdot f(u))_i$, the conditional probability of observing a particular neighborhood $N_s(u)$ around node u given an embedding f can be written as follows:

$$P(N_s(u)|\theta) = \prod_{v_i \in N_s(u)} P(v_i \in N_s(u)|\theta) \tag{8}$$

$$= \prod_{v_i \in N_s(u)} \frac{\exp(f(v_i;\theta) \cdot f(u;\theta))}{\sum_{v_j \in V} \exp(f(v_j;\theta) \cdot d(u;\theta)} \tag{9}$$

Plugging into the objective function above, we can define the simple optimization problem:

$$\theta = \arg\max_{\theta} \sum_{u \in V} \left(-\log Z_u + \sum_{v_i \in N_s(u)} f(v_i;\theta) \cdot f(u;\theta) \right) \tag{10}$$

with the normalizing constant $Z_u = \sum_{v \in V} exp(f(u;\theta) \cdot f(v;\theta)$. The above objective function above, which closely resembles that of softmax regression, can be optimized using stochastic gradient ascent to yield an optimal feature embedding $\theta \in \mathbb{R}^{|V| \times d}$.

In addition to using it for visualizations, we also used Grover *et al.*'s neighborhood embedding algorithm to extend and further corroborate Agrawal *et al.*'s findings that proteins involved in the same disease often share neighborhoods. Specifically, we used the node embedding approach described above to embed the PPI network into a 128-dimensional Euclidean space that preserves networks neighborhood. We then calculated the distance between every pair of proteins in the network using the L2-norm and reported the mean distance: 3.16. Next, we computed the distance between every pair of proteins in the same disease pathway and computed the mean distance for each of our 519 diseases. A small fraction of diseases had mean distance between disease proteins that was smaller than the mean distance between random proteins. Diamond-Blackfan Anemia was one such disease with a mean distance of 1.96. However, as we can see in Figure 4, the majority of diseases

[13] Monica Agrawal, Marinka Zitnik, Jure Leskovec; Large-Scale Analysis of Disease Pathways in the Human Interactome; bioRxiv 189787; doi: https://doi.org/10.1101/189787

[14] Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2016.

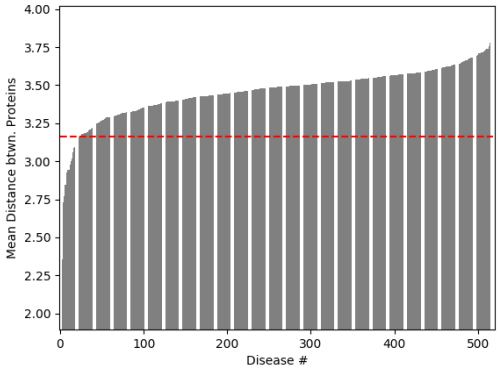


Figure 3: Mean distance between proteins in the same disease pathway for each of 519 disease pathways. The red line indicates the mean distance between randomly selected proteins in the network.

did not exhibit such neighborhood localization. The median distance between proteins in the same disease pathway was 3.49, higher than the mean distance between proteins chosen at random. This confirms the observations made by Agrawal *et al.* that though some diseases show strong neighborhood locality, the majority do not.

Agrawal *et al.*'s findings and our experiment using network neighborhood embedding put into question the widespread existence of clearly defined disease modules. Thus, protein prediction methods that presume the existence of disease modules or rely too heavily on the supposed interconnectivity of disease proteins may be limited in their success.

3.3 Higher-Order Connectivity of Disease Pathways in PPI Network

While many disease pathways may not show strong neighborhood locality in the PPI network, it was recently shown that many disease pathways do show Higher-order network connectivity. Higher-order network structures called network motifs are induced subgraph that occur in a larger network. In Agrawal *et al.*'s analysis, it was found that for many diseases, active proteins tend to take on the same structural role in network motifs of the PPI network. For 60% of diseases studied, proteins in the disease pathway occupy specific positions within certain motifs at a much higher rate than proteins drawn at random. (We call these positions orbits.) [15] As we can see in Figure 3, complex network motifs are particularly overrepresented in disease pathways. This indicates that in a large number of human diseases there exist patterns of higher-order network connectivity that transcend simple edge based connectivity. In other words, we can identify patterns in the structural roles of disease proteins. These observations surrounding

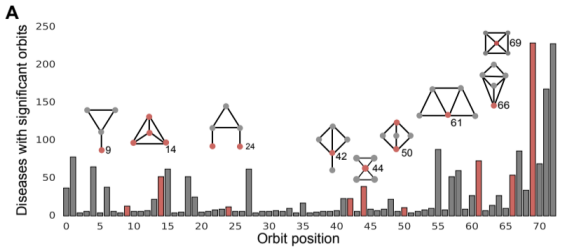


Figure 4: Histogram recording the number of diseases out of 519 analyzed for which disease proteins where overrepresented at each orbit position. Source: Agrawal *et al.*

higher order network patterns in disease pathways provides a promising direction for new methods of disease protein prediction. [16]

In the last two sections, we showed that disease pathways show both edge connectivity and patterns in higher order network structures. Optimal methods of disease protein prediction should simultaneously leverage patterns at both the edge and neighborhood connectivity level as well as the higher-order network structure level. We believe Graph Convolutional Networks are rather well-suited to do just that.

4 Previous Work and Methods in Disease Pathway Prediction

Just over a decade ago, the PPI network was hardly complete enough for meaningful analysis of disease networks, let alone for disease protein prediction. At the time, disease proteins were primarily identified by testing proteins in genomic intervals known to be associated with particular phenotypes.[17]
Recently though, with the growth of the the PPI network, network based methods for disease protein discovery have led the charge. Such network based methods fall into four general categories: linkage methods, modularity methods, diffusion methods, and representation learning methods.

1. **Linkage Methods** Oti et al.'s 2006 method for disease gene prediction is prototypical of linkage methods, which are based on the idea that two proteins that share an edge in the PPI network are likely to be active in the same disease phenotype.[18]

2. **Modularity Methods** These methods go a step beyond linkage methods in that they are guided by the assumption that proteins belonging to the same neighborhoods are more likely to be involved in the

[15] Monica Agrawal, Marinka Zitnik, Jure Leskovec; Large-Scale Analysis of Disease Pathways in the Human Interactome; bioRxiv 189787; doi: https://doi.org/10.1101/189787

[16] Monica Agrawal, Marinka Zitnik, Jure Leskovec; Large-Scale Analysis of Disease Pathways in the Human Interactome; bioRxiv 189787; doi: https://doi.org/10.1101/189787

[17] Barabási, Albert-László, Natali Gulbahce, and Joseph Loscalzo. "Network Medicine: A Network-Based Approach to Human Disease."

[18] Oti M, et al. Predicting disease genes using protein-protein interactions. J Med Genet. 2006; 43:691–698.

same diseases.These methods, thus, find tight clusters containing disease pathways and examine member proteins as candidates for a disease pathway. The robust and effective DIAMOnD system for disease protein prediction is perhaps the best example of a modularity method. [19]

3. **Diffusion Methods** Diffusion methods use the notion of a random walk on a graph (such as those central in the personalized page-rank algorithm) from known disease pathway nodes and rank candidate proteins based on their proximity to known proteins. For example, one can design an rather effective disease protein prediction algorithm that ranks proteins based on the personalized Pagerank vector with a teleport vector seeded with known disease proteins. [20]

4. **Representation Learning Methods** Since 2011, representation learning methods have grown in popularity. Methods in this broad category use latent factors or embeddings that represent protein features relevant to protein-disease classification.

Our approach, applying Graph Convolutional Networks to the PPI network, falls into the representation learning category. Thus, when it comes to assessing the performance of our model, we will look to representation learning methods for comparison. We've implemented an existing representation learning method that uses Grover *et al.*'s neighborhood-preserving network embedding from section 2. Specifically, we run Grover *et al.*'s node2vec with $p = q = 1$ on the PPI network to produce a 128-dimensional embedding that is fed in as feature vector to Logistic Regression for prediction. [21]

5 Graph Convolutional Networks

As we discussed above, several studies have shown that many disease pathways exhibit both neighborhood connectivity and higher-order structure patterns in the protein-protein interaction network.[22] The PPI network, thus, offers us a valuable dataset for disease protein prediction. However, higher-order structure and neighborhood connectivity patterns are hardly explicit in the raw network data: rather, they are latent in the PPI adjacency matrix and have to be uncovered. In order to effectively predict disease proteins with the PPI network, we need a model that can uncover those higher-order structure and neighborhood connectivity patterns from the raw adjacency data. The disease-protein prediction problem, thus, in some ways resembles that of image recognition where image features are deeply latent in a raw RGB regular grid. By learning non-linear combinations of input data, neural networks have been shown to perform effectively on image recognition tasks and other classification problems where patterns of interest are latent in raw input data. It, thus, feels natural to apply neural networks to the disease protein prediction problem, where higher-order connectivity patterns representative of disease are latent in the raw adjacency matrix.

Applying neural networks to network data is hardly trivial, though. Let's consider the naive approach of feeding a network adjacency matrix possibly concatenated with a supplementary feature matrix into a fully-connected deep neural network. This will fail for two reasons: (1) for large networks such as the PPI network, the number of trainable parameters will make backpropagation intractable and would likely lead to overfitting, and (2) the learned model will be spatially dependent – that is it will only be able to learn patterns relative to specific nodes in the network, and thus will be fail to uncover higher-order network structures. As we

5.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) provide an effective model for classification tasks where input data can be naturally represented as a regular grid. Image recognition is one such grid-based classification task, and it has been, of course, emblematic of the recent success of CNNs.[23] The effectiveness of CNNs on image recognition tasks can be attributed to their capacity to learn features independent of spatial location in the grid. This means they can learn to identify the same image features wherever they might appear in the input grid.[24]

CNNs achieve spatial invariance through the use of *convolutional filters*. A convolutional filter is a small grid of learnable weights, roughly analogous to the hidden-layer weights in a vanilla neural network. Generally, the dimensions of a convolutional filter should be substantially smaller than the dimensions of the input grid. In forward-propagation of a vanilla neural network, the activations of one layer are simply dotted with the weights for each of the hidden units in the following layer. In CNN forward-propagation, we convolve (slide) our filters over the entire output grid from the previous layer, isolating small sub-grids, and compute the dot product between the sub-grid and filter weights. Through the convolution or sliding of filters, a filter weights

[19] Ghiassian, Susan Dina, Jörg Menche, and Albert-László Barabási. "A DIseAse MOdule Detection (DIAMOnD) algorithm derived from a systematic analysis of connectivity patterns of disease proteins in the human interactome." PLoS computational biology 11.4 (2015): e1004120.

[20] R. Andersen, F. Chung and K. Lang, "Local Graph Partitioning using PageRank Vectors," 2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06), Berkeley, CA, 2006, pp. 475-486.

[21] Grover, Aditya, and Jure Leskovec. "node2vec: Scalable feature learning for networks." Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2016.

[22] Monica Agrawal, Marinka Zitnik, Jure Leskovec; Large-ScaleAnalysis of Disease Pathways in the Human Interactome; bioRxiv189787; doi: https://doi.org/10.1101/189787

[23] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[24] Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. "Convolutional neural networks on graphs with fast localized spectral filtering." Advances in Neural Information Processing Systems. 2016.

can be updated to identify grid features independent of spatial location. [25]

5.2 Convolutional Neural Networks and Graphs

Above, we discussed two reasons why the naive application of neural networks to graphs is problematic: (1) the number of weights in the neural network will be intractable, and (2) the model will be spatially dependent. The idea of extending Convolutional Neural Networks to graphs is thus an attractive one because it directly addresses those two concerns through weight sharing via convolutions. But applying the idea of convolutional neural networks to graphs is hardly trivial: while "sliding" or convolving a small grid-filter over an image is easy to grasp spatially, it's much less clear how one might "slide" or convolve filter over a graph, or even what a graph filter might look like.

5.3 Convolutional Filters on Graphs

When we apply a $k \times k$ filter in grid-based Convolutional Neural Networks, we compute for each cell in the grid some linear combination of the values in that cell's surrounding $k \times k$ sub-grid using the filters unique weights. In other words, we compute a linear combination of all of the values of the cell's immediate neighbors.

As we demonstrate in Figure 5, we can roughly extend this notion of filtering naturally to graphs given that we have some vector-valued signal on the graph, which we denote $X \in \mathbb{R}^{n \times d}$. A signal on a graph $X \in \mathbb{R}^{n \times d}$ where $X_i \in \mathbb{R}^d$ simply assigns a vector to each node in the graph. If we have some signal X on a graph, the signal on some node X_i is roughly analogous to a vector-value in cell number i of a grid (think RGB vector). When we apply an order k filter to a node in a graph, we compute a linear combination of the signals on each of that nodes k-hop neighbors. More

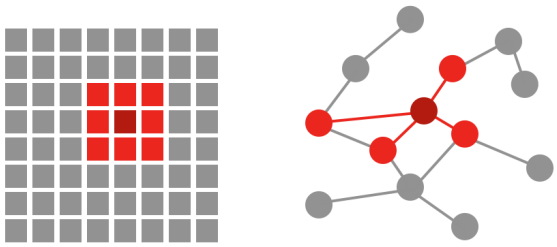


Figure 5: The notion of a filter on a grid, can be intuitively applied to network. On the left, our grid filter sized 3 x 3 (hence k = 3), and on the right our network filter has k = 1, because it computes the one hop neighbors of a central node.

formally, we can define the computation of an order-k filter in the vertex domain as Shuman *et al.* do:

$$f(i) = wX_i + \sum_{j \in N(i,k)} wX_j \tag{11}$$

where $w \in \mathbb{R}^d$ is some weight vector for the signal $X_j \in \mathbb{R}^d$, $N(i,k)$ is the set of k-hop neighbors of i, and x_j is the signal on node j.[26]

5.4 Graph Convolutional Networks

Using this conception of filters on graphs, we can derive a graph convolutional neural network model that was proposed by Kipf & Welling *et al.*[27]. In this section, we present the architecture for the GCN and relate it back to our conception of network filters derived in the previous section. Note that we present the models propagation rules in matrix form — that is, fully vectorized over nodes.

The GCN model will consist of an: **input layer** whose activation consists simply of the identity matrix I (Note: this identity matrix can be replaced with a full feature matrix X, but we only use the identity in our work). We denote this activation $H^{[0]} = I \in \mathbb{R}^{n \times n}$. The model will then consist of L **hidden layers** where $h^{[\ell]}$ denotes the number of units in the ℓ^{th} hidden layer and $H^{[\ell]} \in \mathbb{R}^{n \times h^{[\ell]}}$ is that layer's activation. The ℓ^{th} hidden layer will also be parameterized by a weight matrix $W^{[}\ell] \in \mathbb{R}^{h^{[\ell-1]} \times h^{[\ell]}}$. Furthermore, at each hidden layer will apply some non-linear activation function such as ReLU. Finally, the the neural network model will consist of one **output layer** whose activation is the softmax function applied row-wise.

Nothing about the neural network outline described above differentiates it from a vanilla neural network. However, we can incorporate graph filtering at each of the hidden layers by defining the following forward-propagation for the activation of the ℓ^{th} layer:

$$H^{[\ell]} = \sigma(\hat{A} H^{[\ell-1]} W^{[l]}) \tag{12}$$

where we define $\hat{A} = D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}$ and $A \in \mathbb{R}^{n \times n}$ is the network's adjacency matrix and $D^{-\frac{1}{2}}$ and D is the degree matrix $D_{i,i} = \sum_j (A + I)_{i,j}$. This propagation rule applies $h^{[\ell]}$ filters to the signal from the previous layer of the network. To understand how, let's consider the following element-wise definition of activation of the ℓ^{th} layer:

$$H^{[\ell]}_{i,k} = \sigma\left(\sum_{j=1}^{n} \hat{A}_{i,j} (H^{[\ell-1]} W^{[l]})_{j,k} \right) \tag{13}$$

Taking advantage of the fact that $\hat{A}$ is a close variant of the

[25] Gu, Jiuxiang, et al. "Recent advances in convolutional neural networks." arXiv preprint arXiv:1512.07108 (2015).

[26] Shuman, David I., et al. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains." IEEE Signal Processing Magazine 30.3 (2013): 83-98.

[27] Thomas N. Kipf and (2016). Semi-Supervised Classification with Graph Convolutional Networks. CoRR, abs/1609.02907

adjacency matrix A, we can rewrite Equation 13 as:

$$H_{i,k}^{[\ell]} = \sigma\left(\frac{1}{d_i}(H^{[\ell-1]}W^{[l]})_{i,k} + \sum_{j\in N(i,1)} \frac{1}{\sqrt{d_i d_j}}(H^{[\ell-1]}W^{[l]})_{j,k}\right) \tag{14}$$

where $N(i,1)$ denotes the 1-hop neighborhood of i. Take note of the similarity between Equation 11, the computation of the order-k filter in the vertex domain, and Equation 14 above, the elementwise definition of the activation of the ℓ^{th} layer. Indeed, we can see that the k^{th} element of the activation on node i is an order-1 network filter where the activations of the previous layer $H^{[\ell-1]}$ are the signal on the network, and the columns the weight matrix of the current layer $W^{[l]}$ are the weights for each of the $h^{[\ell]}$ spectral filters. This is evident since $(H^{[\ell-1]}W^{[l]})_{j,k} = H_{j,t}^{[\ell-1]}H_{t,k}^{[\ell-1]} = \sum_{t=1}^{h^{\ell-1}} H_{j,t}^{[\ell-1]}H_{t,k}^{[\ell-1]}$.

5.5 Spectral Filter Motivation

Note that Kipf & Welling present a slightly different motivation for the same forward-propagation rule based on spectral filters in the Fourier-domain. Specifically, they leverage the observation offered by Shuman *et al.* that an order-k filter in the vertex domain is equivalent to a frequency filter in Fourier graph domain. [28] We refer the reader to Kipf & Welling and Hammond *et al.* for an in depth coverage of the spectral filter motivation. [29] [30]

6 Data

We have used and plan on using several different datasets for training and evaluation of our GCN model.

Human Protein-Protein Interaction Network Our first dataset is the protein-protein interaction (PPI) network compiled by Menche *et al.*[31] and Chatr-Aryamontyi *et al.* [32]

Disease Pathway Dataset The second dataset is a listing of 519 diseases and the proteins known to be active in each. These disease-protein associations come from DisGeNET, a centralized database containing information on disease-protein relationships. Across all diseases in the dataset, the median number of associated proteins is 21 and the minimum is 10. Some more complex diseases have over one hundred associations.

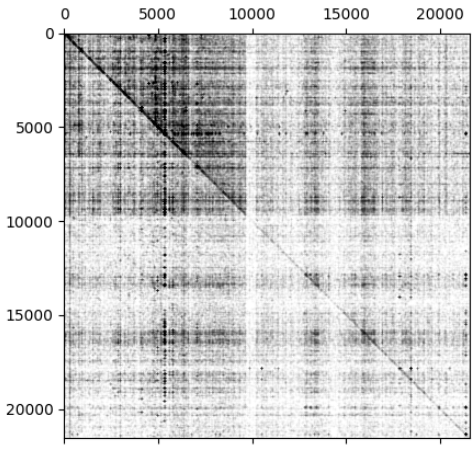


Figure 6: A visual representation of the PPI network adjacency matrix.

7 Experimental Set-up

We ran a disease protein prediction experiment using a graph convolution network for each of the 519 diseases in our dataset. In this section we report the specifics of our experimental set-up.

7.1 The Graph Convolutional Network

The GCN implementation we leveraged was provided by Kipf & Welling, and it directly matches the theoretical description of GCN's provided above.[33] After a brief period of hyperparameter tuning on a small development set of 10 diseases, we configured our GCN with the hyperparameters found in Table 1. In order to isolate performance on the PPI network specifically, we ran the GCN without providing supplementary feature vectors. Thus, we fed an identity matrix to the first layer forward propagation rule:

Table 1: GCN Hyperparameters

Num. Hidden Layers	1
Num. Hidden Units - Layer 1	16
Initial Learning Rate	0.01
Regularization	L2
Regularization Weight	0.005
Max. Chebyshev Poly. Degree	3

$$H^{[1]} = \sigma(D^{-\frac{1}{2}}AD^{-\frac{1}{2}}IW^{[1]}) \tag{15}$$

[28] Shuman, David I., et al. "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains." IEEE Signal Processing Magazine 30.3 (2013): 83-98.

[29] Thomas N. Kipf and (2016). Semi-Supervised Classification with Graph Convolutional Networks. CoRR, abs/1609.02907

[30] Hammond, David K., Pierre Vandergheynst, and Rémi Gribonval. "Wavelets on graphs via spectral graph theory." Applied and Computational Harmonic Analysis 30.2 (2011): 129-150.

[31] Menche, J, Sharma, Amitabh, Kitsak, Maksim, Ghiassian, Susan Dina, Vidal, Marc, Loscalzo, Joseph Barabasi, Albert-Laszlo (2015). Uncovering disease-disease relationships through the incomplete interactome. Science, 347.

[32] Chatr-aryamontri,Andrew et al."The BioGRID Interaction Database: 2015 Update." Nucleic Acids Research 43.Database issue (2015): D470–D478. PMC. Web. 17 Nov. 2017.

[33] Thomas N. Kipf and (2016). Semi-Supervised Classification with Graph Convolutional Networks. CoRR, abs/1609.02907

For each disease, we ran back-propagation on the GCN until convergence.

7.2 A Disease Protein Prediction Experiment

For each of the 519 disease in our dataset, we ran a disease protein prediction experiment using 5-fold cross-validation. That is, for each disease we split the pathway into 5 equally sized sets. On each iteration of cross-validation, we use the proteins in 4 out of the 5 sets as training data and withhold the last set as a test set.

For each train-test split, we formulate a semi-supervised node classification problem as described in section 2 and run our GCN. Specifically, we feed the following training data into our GCN and train with backpropagation on the cross-entropy loss over labeled examples.

1. Adjacency Matrix, A This is our PPI-Adjacency matrix provided in the data section.
2. Label Vector y First, let's call our set of disease proteins d. We label each of these diseases as positive. As described in section 2, to handle the fact that the disease pathway dataset lacks negative labels, we randomly choose a set $\hat{d}$ of unlabeled proteins and label them as negative. For our experiments we kept the number of positive and negative labels equivalent. Using our two sets d and $\hat{d}$, we build a partial label vector y as defined below:

$$y^{(i)} = \begin{cases} 1 & \text{if } i \in d \\ 0 & \text{if } i \in \hat{d} \\ NA & \text{otherwise} \end{cases} \tag{16}$$

3. Feature Matrix, I As described above, we feed the identity matrix as our feature matrix.

For testing, we feed in the same data except that the Label vector is built using the test proteins d_t. We then run forward-propagation to compute predicted proteins. Finally, we compute the performance metrics outlined in the following subsection.

At the end of the 5-fold cross validation we average metrics across the train-test splits.

7.3 Performance Metrics

For each disease we compute the following average metrics across each train-test split in the cross-validation.

1. **Train/Test Accuracy**

   The fraction of labeled proteins that are correctly classified by the GCN. Note that since the negative labels are randomly generated, this is not the most useful metric.

2. **Recall-at-K**

   The fraction of all disease proteins within the first k predicted proteins. We computed recall-at-k for $k = 25$ and $k = 100$. Recall-at-k.

3. **Precision**

   The fraction of correctly predicted disease proteins out of all proteins that our model predicted as disease proteins.

4. **Precision-Recall Curve**

   A curve of *recall* against *precision* computed by incrementally reducing the threshold for positive prediction. In general, as the threshold drops, the recall should rise or stay the same and the precision should drop or stay the same.

8 Results

Our GCN disease protein prediction method shows very strong results when compared to our implementations of other common representation learning methods.

It's worth noting that because the pipeline is trained with the Adam Optimization technique, the results are slightly stochastic but the overall trend is clear.[34]

Table 2: GCN Results (Recall at 100)

Metric	GCN	node2vec
Mean	0.434	0.366
Std. Dev.	0.182	0.237
Median	0.45	0.338
Max	1.0	0.987
Min	0.050	0.014

We put our results into context in a few different ways. First, we present a comparison of our model with recall results from node2vec, which achieves state-of-the-art results with minimal parameter adjustment. Within Figure 7, we graph the relative improvement of our technique on the full disease sets.

We notice that recall-at-100 is low for many diseases, which mirrors the behavior of other models. Indeed, many of the proteins that get low-recall scores with other models are similarly difficult for GCNs yet the GCN tends to outpreform others on a range of previously difficult-to-classify diseases. This explains the bulk of the recall improvements come from improving performance for previously-low-recall diseases. We have labeled some prototypical examples of "difficult," "medium," and "easy" to classify proteins with Figure 8.

[34] Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

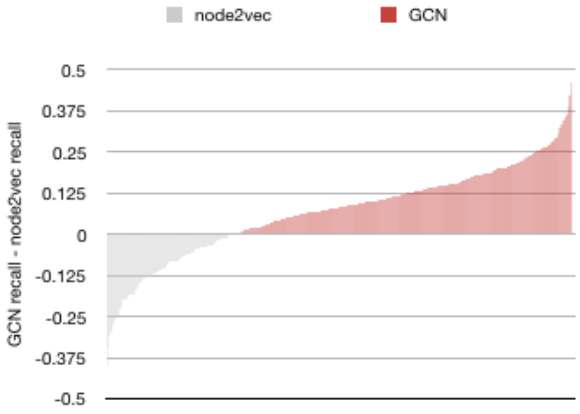


Figure 7: A comparison between the recall-at-100 figures for diseases classified with node2vec and our GCN. Red shows the performance improvements with GCN.

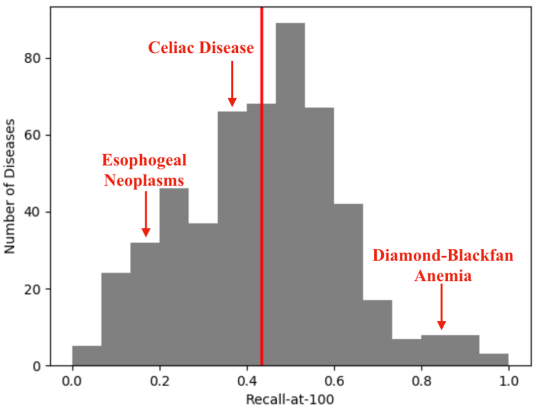


Figure 8: A histogram of the recall-at-100 results for the GCN with exemplatory proteins indicated.

9 Future Work

9.1 Feature Selection

As we discussed above, Graph Convolutional Networks take as input both a graph specified by an adjacency matrix A as well as a feature matrix X which defines a feature vector for each node in the graph. One of the main advantages of Graph Convolutional Networks over other graph clustering/classification techniques is that it works on outside feature information on top of the information encoded in the graph's edges. The question is: what is the best information to encode in the feature vector?

Raza *et al.* in their paper "Protein Features Identification for Machine Learning Based Prediction of Protein-Protein Interactions" discuss the use of the Gene Ontology for population of a protein's feature vector.[35] The Gene Ontology provides detailed information on protein structure and function within many different species. We plan on building short feature vectors $x^{(i)}$ that consist of one-hot encodings on a set of high-level functional classes.[36]

9.2 Model Tuning

Due to the train times required for a robust model, we forwent performing gridsearch for hyperparameters. That said, we hypothesize that we can yield significant performance gains by customizing our model more specifically to our task. Two areas for easy customization are increasing the depth of layers and count of hidden nodes.

Additionally, in the section on Higher-Order Network Structure, we discussed how some node and edge orbits are over-represented in the protein-protein interaction network. If time permits, we plan on running our GCN on a PPI network weighted by overrepresented higher-order network structures.

[35] Raza, Khalid (2017). Protein Features Identification For Machine Learning-Based Prediction Of Protein-Protein Interactions.

[36] https://www.nature.com/articles/nrg1272