LEARNING ON GRAPHS: SUPERVISED AND UNSUPERVISED METHODS

Scott Emmons

A thesis submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Bachelor of Science with Honors in the Department of Mathematics.

Chapel Hill
2019

Approved by:

Shankar Bhamidi

Jason Metcalfe

Peter J. Mucha

ABSTRACT

SCOTT EMMONS
Learning on Graphs: Supervised and Unsupervised Methods
(Under the direction of Dr. Peter J. Mucha)

We study two methods for learning from network graph data. First, we present a novel method for the unsupervised learning problem of community detection. The proposed method is, to the best of our knowledge, the first enabling users to "zoom in" and "zoom out" on communities with varying levels of focus on network metadata. Second, we review *Decagon*, a system proposed by Zitnik *et al.* (2018) for the supervised learning task of link prediction. On a biomedical benchmark dataset, *Decagon* achieves state-of-the-art prediction accuracy. This work adds to the network scientist's machine learning toolkit, illustrating its power in a biomedical domain with significant public health impact.

To all my friends over the past four years who have patiently listened to me explain, “Take a piece of paper. Draw a circle on the page for each account on Facebook, and connect the circles with lines if they are Facebook friends. That’s a graph.”

ACKNOWLEDGEMENTS

We are grateful to Zachary Boyd, Peter Diao, Ryan Gibson, and William Weir for discussions throughout the creation of this work. We thank Daniel Edler, Martin Rosvall, and the entire team at the Integrated Science Lab of Umeå University for their collaboration incorporating our work into the Infomap software package. Research reported in this publication was supported by the James S. McDonnell Foundation 21st Century Science Initiative - Complex Systems Scholar Award grant #220020315. The content is solely the responsibility of the authors and does not represent the official views of the sponsor.

This research uses data from Add Health, a program project directed by Kathleen Mullan Harris and designed by J. Richard Udry, Peter S. Bearman, and Kathleen Mullan Harris at the University of North Carolina at Chapel Hill, and funded by grant P01-HD31921 from the Eunice Kennedy Shriver National Institute of Child Health and Human Development, with cooperative funding from 23 other federal agencies and foundations. Special acknowledgment is due Ronald R. Rindfuss and Barbara Entwisle for assistance in the original design. Information on how to obtain the Add Health data files is available on the Add Health website (http://www.cpc.unc.edu/addhealth). No direct support was received from grant P01-HD31921 for this analysis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In its most basic form, a graph, also called a network, is comprised of things and connections between those things. Formally, we call the things "nodes" or "vertices," and we call the connections between them "edges." Mathematically, we represent a graph $G = (\mathcal{V}, \mathcal{E})$ by a set of vertices $\mathcal{V}$ and a set of pairwise edges $\mathcal{E} = (v_1, v_2)$ with $v_1, v_2 \in \mathcal{V}$.

Networks model many real-world systems. The students in a high school form a friendship network. The proteins in cells form interaction networks. The roadways in a country form a transportation network.

For different kinds of systems, we can extend the mathematical model above. We might model airline flights with a *directed* network so that an edge $(v_1, v_2) \in \mathcal{E}$ encodes an asymmetric relationship, a flight departing from v_1 and traveling to v_2, and we might assign a scalar value to the edge for the distance traveled, forming a *weighted* network. Our networks could be *multimodal*, with different types of edges, such as roadways, train routes, and flights, connecting different types of nodes, such as towns and cities, in the network. And the nodes and edges in our network could be *temporal*, appearing and disappearing over time.

Many types of network analyses exist, ranging from modeling dynamics on networks – how might disease spread in a social network? – to understanding networks robustness – how many electricity lines must fail before the city is without power? In this work, we focus on two machine learning tasks in networks, *community detection* and *link prediction*.

From our everyday experience, we intuit that networks have groups of community structure. In our friendship network, for example, we have groups of friends from our hometown, groups of friends from school, and groups of friends from work. Community detection is an unsupervised learning task that seeks automatically to detect these sorts of latent clusters in a graph, and it has

widespread application. For example, neuroscientists have applied community detection methods to study the functionality of the human brain, and social scientists have applied community detection methods to study the spread of information on social networks.

In chapter 2, we present a novel method for community detection in the presence of node metadata. Our algorithm allows users explicitly to specify the relative weight of node metadata, such as grade or sex in a high school social network, to guide the community detection process. Whereas previous methods for community detection with metadata assume that the user only cares about the node metadata insofar as it statistically correlates with edge formation on the graph, our method allows users to choose how much to overfit the network partition to the metadata, to "zoom in" and "zoom out" on partitions with varying levels of metadata focus.

Our everyday experience also gives us intuition about link prediction. If we have ever had Amazon recommend us a product to buy, Netflix recommend us a movie to watch, or Facebook recommend us someone to add as a "friend," we have been the subject of a link prediction algorithm. Any time that we want to know missing or future edges in a graph, we are interested in the supervised learning problem of link prediction.

In chapter 3, we review *Decagon*, a state-of-the-art link prediction algorithm (Zitnik et al., 2018). *Decagon*'s development is grounded in its application to a multimodal polypharmacy network of drugs and proteins. An edge between two drugs in the network represents a polypharmacy side effect, a side effect from taking two drugs concurrently that is not the cause of either drug alone. As drug-related problems cost the United States $177 billion a year (Ernst and Grizzle, 2001) and 47 million Americans are under polypharmacy treatment (Kantor et al., 2015), *Decagon* demonstrates how learning algorithms for graph data can have significant public health impact.

After studying a particular community detection algorithm and link prediction method up close, we conclude in chapter 4 by stepping back and commenting in general on the role of supervised and unsupervised learning methods in directions for future research.

CHAPTER 2

AN UNSUPERVISED LEARNING METHOD FOR COMMUNITY DETECTION*

2.1 Background

As network science has found application in a variety of real-world systems, ranging from the biological to the technological, so too has community detection in networks received widespread attention (Porter et al., 2009; Fortunato, 2010; Fortunato and Hric, 2016; Shai et al., 2017). Traditionally, community detection methods have focused solely on the topology of the network, optimizing an objective function defined on the network structure that captures a particular notion of community, such as intra-community edge density and inter-community edge sparsity. Many approaches, ranging from the statistical to the information theoretical, have been used for community detection, and tradeoffs between these approaches include describing extant links versus predicting missing links (Ghasemian et al., 2018).

More recent community detection work utilizes node metadata such as the grade and gender of students in a high school social network. As the No Free Lunch theorem states, community detection algorithms must make tradeoffs (Peel et al., 2017), and node metadata can be used to guide community detection. For example, Newman and Clauset demonstrated that their stochastic block model approach can choose either to partition a middle school and high school social network into communities by grade or into communities by race, depending on the metadata of interest (2016). Similarly, Hric *et al.* (2016) developed an attributed SBM from a multilayer perspective, with the attribute layer modeling relational information between attributes. Stanley *et al.* (2018a) considered a different graphical model relating connections and attributes, with assumptions on the

*The content of this chapter is currently under review for publication in *Physical Review E*.

attribute distributions, to develop a stochastic block model with multiple continuous attributes. The I-louvain method (Combe et al., 2015) extends the well-known Louvain algorithm (Blondel et al., 2008) for modularity maximization by including attributes in their "intertia-based modularity." Yang *et al.* proposed CESNA (2013) and He *et al.* proposed CNMMA (2018) to identify communities by learning a latent space that generates links and attributes. Peel *et al.* (2017) established a statistical test to determine if attributes correlate with community structure, and they developed an SBM with flexibility in how strongly to couple attributes and community labels in the corresponding stochastic block model inference. In related work, Stanley *et al.* (2018b) propose a test statistic based on label propagation for the alignment of node attributes with connectivity patterns.

Prior work, such as the method of Newman and Clauset, assumes that one only cares about the node metadata insofar as that metadata explains network formation (2016). For example, a social science researcher who cares about gender groups in a high school social network has no way to communicate to the algorithm of Newman and Clauset that she has a special interest in the gender metadata; the algorithm will use the gender metadata insofar as it explains network formation, and otherwise the algorithm will ignore it. A key motivation for our work is that a network analyst might be specially interested in how a particular metadata type describes existing network structure regardless of its role in generating that structure. With our method, the above researcher can specify exactly how much more she weights communities describing gender to communities describing network structure.

Ghasemian *et al.* (2018) characterize this tradeoff between a statistical model of network formation, given by the algorithm of Newman and Clauset (2016), and an information-theoretic description of observed structure, given by the map equation (Rosvall and Bergstrom, 2008), as a tradeoff between under- and overfitting in community detection. From the point of view of this framework, our method's contribution is enabling users to choose how much to overfit the metadata in describing the observed network structure.

Most closely related to our approach is the content map equation proposed by Smith *et al.* (2016). The content map equation, as we later describe in more detail, adds an additional term to the map equation (Rosvall and Bergstrom, 2008) that introduces entropy based on the metadata. This modification to the map equation encourages intra-module homogeneity of node metadata values.

Our work extends the content map equation, categorizing the different sources of entropy in the map equation into the "inter-module codebook," "intra-module codebooks," and "metadata codebooks." Within this framework, we introduce a tuning parameter to the metadata codebooks that allows explicit specification of the relative importance of particular metadata types. Similar to how focusing knobs are an essential feature of a microscope, adding a tuning parameter to the content map equation is essential to its function, allowing one to "zoom in" and "zoom out" on communities with varying levels of focus on the metadata.

2.2 Methodology

The map equation frames the problem of community detection as minimizing the description length of a random walk on the network (Rosvall and Bergstrom, 2008). In developing a code to compress the description of the random walk, the map equation necessitates that each codeword corresponds to an identifiable entity in the graph. It designates codewords for hard partitions of nodes into modules, codewords for individual nodes, and codewords for a special "exit" keyword for each module. As the codeword for a given node needs only to be unique within that node's module, the module names and node names function like geographic city names and street names. The output of the map equation is a sort of "map," optimized for data compression, that captures patterns in the data.

The map equation's entropy arises from two different types of codebooks. The "inter-module codebook," consisting of module names, describes movement between modules. The "intra-module codebooks," consisting of node names and special "exit" codewords, describe movement within modules. The sum of the entropies of these codebooks, weighted by their relative frequencies, gives the per-step average number of bits needed to describe an infinite random walk on the network for a given partition M of the nodes into m modules:

$$L(\mathrm{M}) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i).$$

Here, $H(\mathcal{Q})$ is the entropy of the inter-module codebook, used with relative frequency $q_{\curvearrowright}$, and $H(\mathcal{P}^i)$ is the entropy of the intra-module codebook for module i, used with relative frequency $p^i_{\circlearrowright}$.

The traditional map equation is concerned solely with topology; only the path of the random walker must be encoded. To extend the map equation to networks annotated with metadata, *we additionally require that the value of the metadata at each step of the random walk be encoded.* The game of the encoder is to encode at which node a random walker is at each step of the random walk. Like in the traditional map equation, the encoder must report whenever the random walker changes modules. Additionally, the encoder must report the value of the metadata at each step of the random walk. We require that the metadata values be encoded uniquely within each module, a requirement that, as we will later see, favors network partitions in which module labels align with metadata labels.

To model network dynamics, we consider a random surfer on the network. With probability $1 - \tau$, the surfer behaves like a random walker, choosing to walk along an outgoing edge of its current node with probability proportional to the outgoing edge weights. With probability τ, the surfer teleports to an arbitrarily chosen node selected uniformly at random in the network. Although unnecessary in undirected networks, teleportation guarantees desirable properties of the random walk in directed networks such as not becoming stuck at a node with no outgoing edges. Considering this surf in the limit of an infinite number of steps, we arrive at a steady state distribution p_α for every node α in the network. For notational convenience, we normalize the outgoing edge weights of every node α so that $\sum_\beta w_{\alpha\beta} = 1$. We let U denote a finite, discrete set of all metadata labels and assume that each node α is tagged with exactly one $u_\alpha \in U$.

The content map equation models entropy generated by the random surfer's movements between modules and within modules identically to the traditional map equation. Between modules, we encode whenever the random surfer exits one module and enters another module. The chance at any given step that the surfer exits module i is

$$q_{i\curvearrowright} = \tau \frac{n - n_i}{n - 1} \sum_{\alpha \in i} p_\alpha + (1 - \tau) \sum_{\alpha \in i} \sum_{\beta \notin i} p_\alpha w_{\alpha\beta},$$

where n_i is the number of nodes in community i. We denote the total chance at any given step that the random surfer exits a module as

$$q_{\curvearrowright} = \sum_{i=1}^{m} q_{i\curvearrowright}.$$

By Shannon's source coding theorem, the minimum entropy to encode the transitions between modules, the encoding we call the "inter-module codebook," is

$$H(\mathcal{Q}) = -\sum_{i=1}^{m} \frac{q_{i\curvearrowright}}{q_{\curvearrowright}} \log(\frac{q_{i\curvearrowright}}{q_{\curvearrowright}}).$$

The random surfer's movement within modules is another source of entropy in the map equation. Within each module, we encode the name of each node α that the random surfer visits with steady-state frequency p_α, and we use a special "exit" keyword occurring with frequency $q_{i\curvearrowright}$ to encode when the random surfer exits the module. Together, these terms give the intra-module entropy for module i weight

$$p^i_{\circlearrowright} = q_{i\curvearrowright} + \sum_{\alpha \in i} p_\alpha.$$

By Shannon's source coding theorem, the minimum entropy to encode the transitions within a module, an encoding we call an "intra-module codebook," is

$$H(\mathcal{P}^i) = -\frac{q_{i\curvearrowright}}{p^i_{\circlearrowright}} \log\left(\frac{q_{i\curvearrowright}}{p^i_{\circlearrowright}}\right) - \sum_{\alpha \in i} \frac{p_\alpha}{p^i_{\circlearrowright}} \log\left(\frac{p_\alpha}{p^i_{\circlearrowright}}\right).$$

The content map equation additionally models the entropy of the node metadata values at each step of the random surf. Within each module i, we assign a codeword to each metadata value $u \in U$ that occurs with frequency

$$r^i_u = \sum_{\substack{\alpha \in i, \\ u_\alpha = u}} p_\alpha,$$

and we let the total metadata weight of module i be

$$r^i_{\circlearrowright} = \sum_{u \in U} r^i_u = \sum_{\alpha \in i} p_\alpha.$$

By Shannon's source coding theorem, the minimum entropy to encode the metadata values within module i, in that module's "metadata codebook," is

$$H(\mathcal{R}^i) = -\sum_{u \in W} \frac{r^i_u}{r^i_{\circlearrowright}} \log\left(\frac{r^i_u}{r^i_{\circlearrowright}}\right).$$

By encoding the metadata values separately within each module, we reward partitions whose module labels align with the metadata values. Under this encoding method, if all nodes in a module have the same metadata value, the module name in the inter-module codebook alone thereby fully specifies the metadata value at each within-module movement step, and the metadata codebook contributes zero additional entropy for this module.

Summing the entropies of the inter-module codebook, the intra-module codebooks, and the metadata codebooks, weighted by their frequency of use, the corresponding content map equation for a given partition M becomes

$$L(\mathrm{M}) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{m} p^i_{\circlearrowright} H(\mathcal{P}^i) + \eta \sum_{i=1}^{m} r^i_{\circlearrowright} H(\mathcal{R}^i) \tag{2.1}$$

where we introduce the parameter η to control the relative weight of the metadata entropy. By increasing η, we increasingly favor communities of nodes with shared metadata values. The special case $\eta = 1$ is identical to the method proposed by Smith *et al.* (2016). When each module contains only a single distinct metadata label, or when $\eta = 0$, the corresponding map equation reduces to the traditional map equation.

As one way to interpret η, consider sending our message encoding the random surf over two different discrete channels, one containing the information of the traditional map equation and the other containing the metadata information. If we suppose that there are different costs to access the two channels, we can interpret η as the relative cost to access the discrete channel of metadata information. In this interpretation, η is an *ad hoc*, relative penalty; we are not deriving η from the dynamics of the random surf. By itself, the metadata channel does not contain useful information because the metadata codewords are module-dependent. For all finite values of η, however, the entropy of the traditional map equation contributes to our objective function, and we can assume that the receiver has access to the module information.

One can imagine other ways to extend the map equation with metadata. As one example, instead of requiring that the encoder report the metadata value at each step of the random walk, one might only require the encoder to report when the metadata value changes. Rather than penalizing entropy in the metadata composition of a module, as our framework does, such a formulation would penalize the entropy of neighboring nodes that have different metadata values. As another example, instead

of partitioning the nodes of the network, one might instead partition the edges of the network. An analogous coding game could be played with attributed edges to identify hierarchical and overlapping community structure; for example, see the edge partitioning methods of Ahn *et al.* (2010) and Kim and Jeong (2011). We leave the study of alternative extensions of the map equation with metadata to future work.

Throughout this work, we compare the similarity of partitions with the scikit-learn implementation of adjusted mutual information (AMI) (Pedregosa et al., 2011) by using the following measure proposed by Vinh *et al.* (2010):

$$\text{AMI} = \frac{I(X,Y) - E\{I(X,Y)\}}{\max\{H(X), H(Y)\} - E\{I(X,Y)\}}.$$

We use AMI because it adjusts the observed mutual information, $I(X,Y)$, between partitions X and Y by that expected at random under a hypergeometric model, $E\{I(X,Y)\}$. Normalizing by the expectation-adjusted maximum of the partitions' individual entropies, $H(X)$ and $H(Y)$, the AMI has an expected value of 0 when partitions are random and a maximum value of 1.

Our extended content map equation has been incorporated into the map equation optimization software package Infomap v1.0 (Edler and Rosvall, 2019), which we use for all of our experiments. Because teleportation can blur a network's modular structure (Lambiotte and Rosvall, 2012), we avoid teleportation by modelling all of our networks as undirected and setting the teleportation probability $\tau = 0$.

2.3 Synthetic Graph Results

To analyze how varying η impacts the content map equation's ability to detect communities, we construct synthetic graphs according to a two-block planted-partition stochastic block model (SBM) with $N = 200$ nodes evenly divided into 2 communities, where an edge connecting two nodes in the same community exists with probability p_{in}, and an edge connecting two nodes in different communities exists with probability p_{out}. We additionally annotate each node with one of two discrete attribute labels based on the "noise" parameter. Each node with probability $1-$ noise is assigned an attribute label equating to its community assignment and with probability noise is assigned the

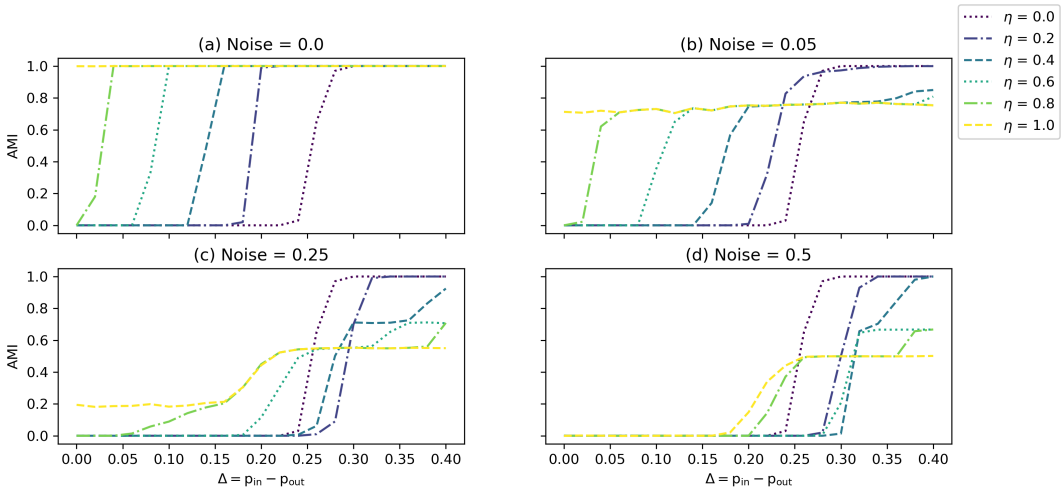

Figure 2.1: Detectability experiments on a planted-partition stochastic block model with $N = 200$ nodes evenly divided into two communities. Each node's attribute label corresponds with its planted partition community with probability noise, and Δ measures the assortativity of the planted partition. We see that focusing on the metadata by increasing η enables the algorithm to overcome the detectability limit when the metadata has strong signal, but increasing η ceilings the algorithm's performance when the metadata is noisy.

opposite attribute label. With noise of 0, the communities and attributes correspond perfectly, and with noise of 0.5, the attributes are totally random. Figure 2.1 shows results for different η exploring the AMI between the planted partition and that identified by the content map equation, where each data point is the average of 100 trials with edge density $\rho = (p_{in} + p_{out})/2 = 0.2$. We show a maximum value of $\eta = 1$ because it is the same as the results for higher values of η.

For the corresponding unannotated SBM, it has been shown in the limit as $N \to \infty$ that the two-block planted-partition structure becomes undetectable for $\Delta = p_{in} - p_{out}$ below the threshold given by $N\Delta^* = \sqrt{4N\rho(1-\rho)}$. (For more detail, see (Decelle et al., 2011; Nadakuditi and Newman, 2012) and the discussion including non-sparse and multilayer networks in (Taylor et al., 2016).) Communities are only detectable when $\Delta > \Delta^*$ because otherwise the community structure is too weak relative to the background noise of the generative model. For the parameters of our experiment, $\Delta^* \doteq 0.063$. The detectability of partitions, however, is distinct from resolution selection, i.e., determining the size of partitions. Experimentally, we find that Infomap, partitioning solely by network structure with $\eta = 0$, transitions from returning a single-community partition at $\Delta = 0.20$ to returning the planted two-community partition at $\Delta = 0.28$. For the remainder of the discussion, we refer to the $\Delta = [0.22, 0.30]$ region as the "selection threshold".

In the presence of metadata signal below the selection threshold, we find as expected that increasing η increases AMI. Although the communities are undetectable from the network connectivity alone, the metadata provides additional information. Moreover, as the metadata becomes more aligned with the communities, it provides a greater boost to the algoirthm's performance. For example, as Fig. 2.1 illustrates, reducing the noise from 0.25 to 0.05 increases the average AMI at $\eta = 1$ from around 0.2 to around 0.7

Our experiments show both that increasing η can benefit AMI when the community structure has relatively low community assortativity, i.e., when Δ is small, and that increasing η can hurt AMI when the communities have relatively high assortativity, i.e., when Δ is large. When Δ is small, increasing η allows the algorithm to detect the signal present in the metadata, which is greater than that present in the network structure. But when Δ is large, increasing η too much causes the algorithm to overfit the metadata and miss the communities present in the network structure. This effect can be seen in Fig. 2.1 at noise 0.25, where below the selection threshold high values of η achieve average AMI of 0.2 compared to average AMI of 0 for low values of η, while above the selection threshold high values of η have average AMI capped at 0.7 while low values of η achieve average AMI approaching the perfect score, 1.

Perhaps surprisingly, increasing η increases AMI below the selection threshold even when the metadata is totally random, i.e., when noise $= 0.5$. In this case, increasing η acts as an effective resolution parameter. By encouraging partitions with homogeneous metadata values, increasing η makes the algorithm prefer a partition containing several smaller communities over the larger, single-community partition. As a result, increasing η below the selection threshold moves the algorithm away from the single-community partition, which has an AMI of 0, to a partition of several communities with positive AMI.

2.4 Real-World Graph Results

2.4.1 Lazega Lawyers Networks of Law Firm Relationships

The Lazega lawyers networks consist of 71 lawyers at a corporate lawfirm in the American Northeast (Lazega, 2001). Surveys were conducted to form the basis of three networks connecting the same actors: the coworking network, based on a survey question asking each lawyer with whom in the

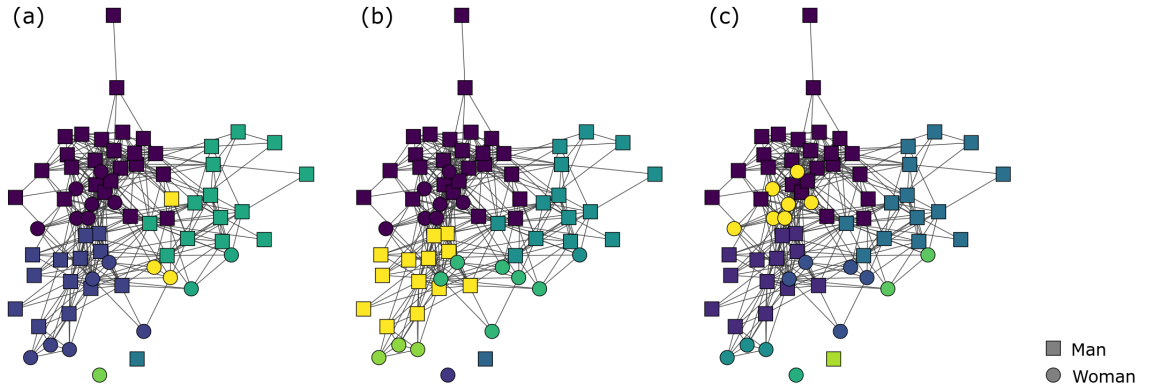

Figure 2.2: The Lazega lawyers friendship network partitioned with the metadata attribute gender at (a) $\eta = 0.0$, (b) $\eta = 0.1$, and (c) $\eta = 0.3$. Color encodes the partition while shape encodes the metadata.

firm the lawyer has worked; the advice network, based on a survey question asking each lawyer to whom in the firm the lawyer has gone for professional advice; and the social network, based on a survey question asking each lawyer with whom in the firm the lawyer socializes outside of work. As node metadata, we additionally use information that each lawyer reported about the lawyer's status (partner or associate), gender (man or woman), office (Boston, Hartford, or Providence), practice (litigation or corporate), and law school (Harvard / Yale, University of Connecticut, or other).

Figure 2.2 illustrates how increasing η affects the returned network partition. The figure shows communities in the friendship network using the metadata attribute gender. Node shapes encode metadata values while node colors encode the algorithm's partition. At $\eta = 0$, the algorithm optimizes for the traditional map equation, returning a partition based solely on network topology. With increasing η, the algorithm returns modules more aligned with the metadata. For example, moving from $\eta = 0$ to $\eta = 0.1$, the man in a module of one man and two women merges into a larger module with many more men, and the two women form a new module with four other women. At $\eta = 0.3$, the modules are either all-male or all-female, and the metadata codebooks contribute zero additional entropy to the content map equation. Note, however, that even as the algorithm increasingly takes the metadata into consideration with increasing η, the algorithm still respects the topology of the network because the random walker proceeds independently of node metadata values.

Figure 2.3 shows the sum of the entropies of each codebook type, weighted by frequency of use but not relatively weighted by η, for partitions of the Lazega lawyers networks at varying η

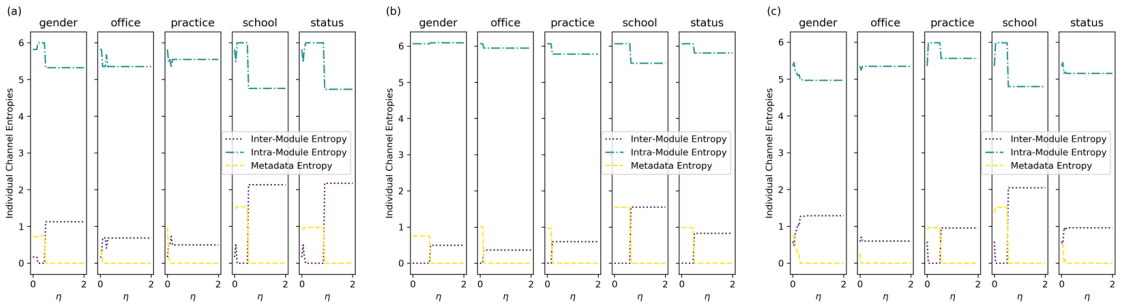

Figure 2.3: Sums of various types of entropy when partitioning the Lazega lawyers (a) coworking, (b) advice, and (c) friendship networks. The sums are weighted by frequency of codebook use but not weighted by η.

for the different metadata types. "Inter-module entropy" measures the first term of Equation 2.1, "intra-module entropy" measures the second term of Equation 2.1, and "metadata entropy" measures the third term of Equation 2.1, unweighted by η. In all the plots, barring a few exceptions due to Infomap's stochasticity, metadata entropy is at its maximum when $\eta = 0$ and decreases until the metadata entropy becomes 0 for sufficiently large η.

As Fig. 2.3 illustrates, once the relative weight of the metadata codebook is sufficiently large, an optimal partition's metadata codebook will necessarily have zero entropy. Optimizing the content map equation in the limit as $\eta \to \infty$ becomes a constrained optimization of the traditional map equation. A candidate optimal partition must have only one metadata attribute per module, and the optimal partition is the partition from this constrained region of partition space optimizing the traditional map equation.

The Lazega lawyers networks, which share the same set of attributed nodes but have different edge types, allow us empirically to study how different edge formation processes influence metadata community detection. In Fig. 2.3b and Fig. 2.3c, the gender panel shows how different connectivity patterns among the same set of attributed nodes can lead to qualitatively different behavior with increasing η. In the Fig. 2.3b advice network, increasing η results in a sharp transition from the topological partition at $\eta = 0$ to the partition with zero metadata entropy that is optimal as $\eta \to \infty$; the algorithm finds no intermediate partitions. In the Fig. 2.3c friendship network, on the other hand, the transition is more gradual. As the algorithm transitions from $\eta = 0$ to the limit as $\eta \to \infty$, it returns multiple intermediate partitions such as the one shown in Fig. 2.2b.

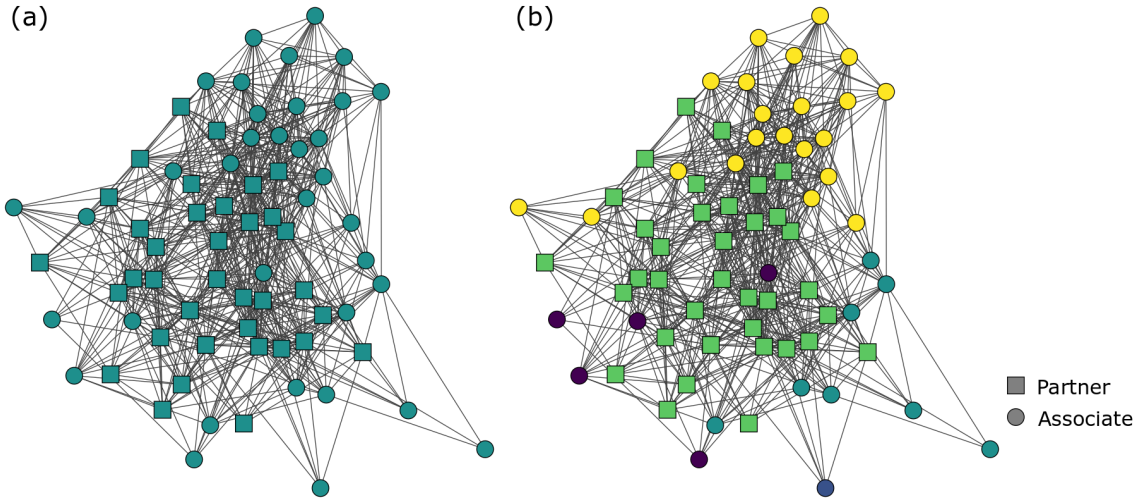

Figure 2.4: The Lazega lawyers advice network partitioned with the metadata attribute status at (a) $\eta = 0$ and (b) $\eta = 0.5$. Color encodes the partition while shape encodes the metadata.

Each panel of Fig. 2.3 summarizes the entropies of an entire set of partitions that can be studied in more depth. For example, consider Fig. 2.4, which shows partitioning the Lazega lawyers advice network with metadata about each lawyer's status in the firm, either partner or associate. Partitioning only on network topology in Fig. 2.4a at $\eta = 0$, the algorithm returns a partition with only module. But in Fig. 2.4b at $\eta = 0.5$, we see the best partition of the network that puts partners and associates into separate modules. While there is one module of partners, there are four modules of associates. In other words, when we constrain the map equation to modules of all partners and all associates, the best description of the flow of advice in the network has one module of partners and four modules of associates.

Perhaps the partners of the firm, who have presumably been around the longest, have spent enough time together that each partner trusts the other partners for professional advice, whereas the associates of the firm have not yet developed trust with all the other associates. Or perhaps the partners of the firm are the most knowledgeable about the firm's operations and form a core module of nodes in the advice network with the associate modules at the periphery. Whatever the cause of the difference between the number of partner and associate modules, the difference is a structure in the network that motivates follow-up study and can only be seen by partitioning with node metadata.

2.4.2 Add Health Network of High School Friendship

The high school friendship network used here results from the US National Longitudinal Study of Adolescent Health and was provided by the Add Health project of the Carolina Population Center. Each of the 795 nodes of the graph is a student in an American middle school (7-8th grade, 12-14 years of age) and corresponding high school (9-12th grade, 14-18 years of age). Edges between nodes represent friendships determined by survey. As metadata for each node, we use the student survey data of grade (range 7-12), race ("white only", "black only", "any Hispanic", "Asian only," or "mixed / other"), school code (middle or high school), and sex (male or female).

The presence of various metadata types allows us to highlight a key feature of the algorithm, that it allows tuning η to see how the network partitions under a particular metadata type of interest. In prior work on community detection with metadata, the method of Newman and Clauset (2016) was applied to the network three times, separately using grade, race, and sex, in each case partitioning the network into two communities. Using grade metadata, the algorithm splits the network into clear middle school and high school groups. Similarly, the algorithm divides the network into a predominantly white and a predominantly black group when it uses race metadata. However, when asked to use sex metadata, the algorithm of Newman and Clauset ignores the sex metadata because it does not have a strong enough correlation with the network structure. As Newman and Clauset note, for someone interested only in the metadata to the extent that it correlates with network structure, it is advantageous for the algorithm to disregard metadata that does not correlate.

But suppose that *a priori* a network analyst knows she cares about a particular metadata type. In our example, a social science researcher might be interested in how the high school friendship network organizes by sex, however strong or weak the sex partition might be. Using the algorithm of Newman and Clauset, there is no way for such a researcher to convey to the algorithm this preference for the sex metadata type. A key feature of our metadata map equation is the ability, using η, to specify the relative weights of a given metadata type compared to the network topology in assigning communities.

Figure 2.5 demonstrates how our algorithm can specify a relative weighting for various metadata types. When $\eta = 0$, all of the partitions follow only the network topology. In that case, our results, consistent with those of the algorithm of Newman and Clauset, show that the metadata attributes

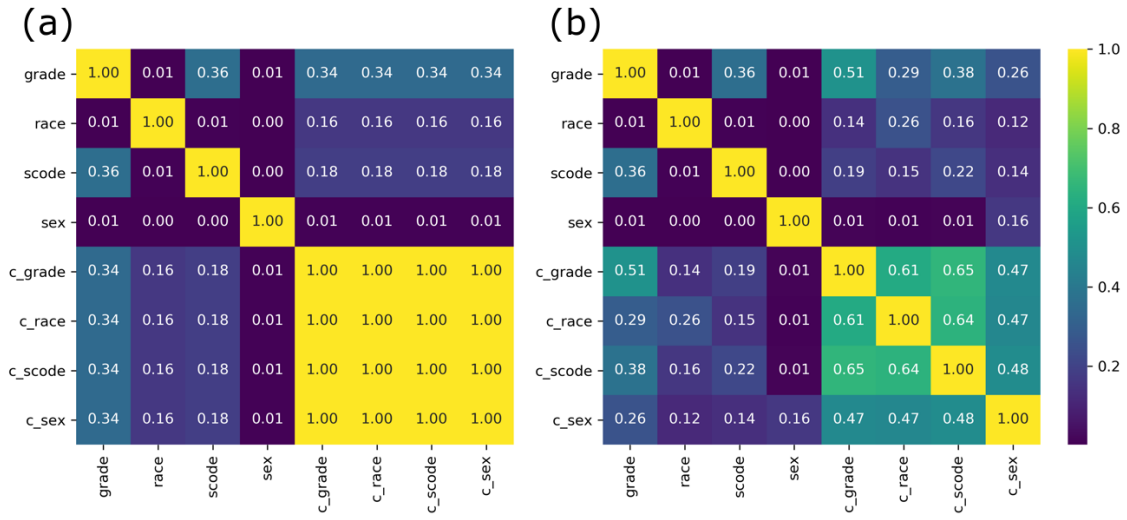

Figure 2.5: Pairwise AMIs of High School Social Network partitions at (a) $\eta = 0$ and (b) $\eta = 0.5$. For example, "grade" is the partition given by each node's grade, and "c_grade" is the algorithm's returned partition when partitioning with the grade metadadta.

of grade, school code, and race have the highest mutual information with the topological partition, with respective AMI values of 0.34, 0.16, and 0.18, while the metadata attribute of sex has the least mutual information with the topological partition, with an AMI of 0.01. When we increase η to $\eta = 0.5$, we see using each of the metadata values (grade, race, school code, sex) that the algorithm finds partitions of the network that, compared to the community detection done with only the network topology at $\eta = 0$, has increased AMI with the metadata.

Importantly, the partitioning of the high school network with a relative metadata channel weight of $\eta = 0.5$ does not simply ignore the network structure. Consistent with the results of Newman and Clauset, we see that grade is the metadata value for which we can achieve the highest AMI between the algorithm's partition and the node metadata, with an AMI of 0.51, and we find that our algorithm's partition using sex has the least correspondence with the node metadata, an AMI of 0.16.

Figure 2.6 illustrates the role of η and the metadata in the community detection process. Each point on the graph of Fig. 2.6 is an AMI calculation. "Grade", "race," "scode" (school code), and "sex" are the partitions of the network given by the respective metadata labels, and "c_grade," "c_race," "c_scode," and "c_sex" are partitions returned by the algorithm given the corresponding metadata type as input and the value of η indicated by the x-axis. The lines of Fig. 2.6 show how the AMIs of pairs of these partitions change with η. Pairs of partitions determined solely by metadata are constant

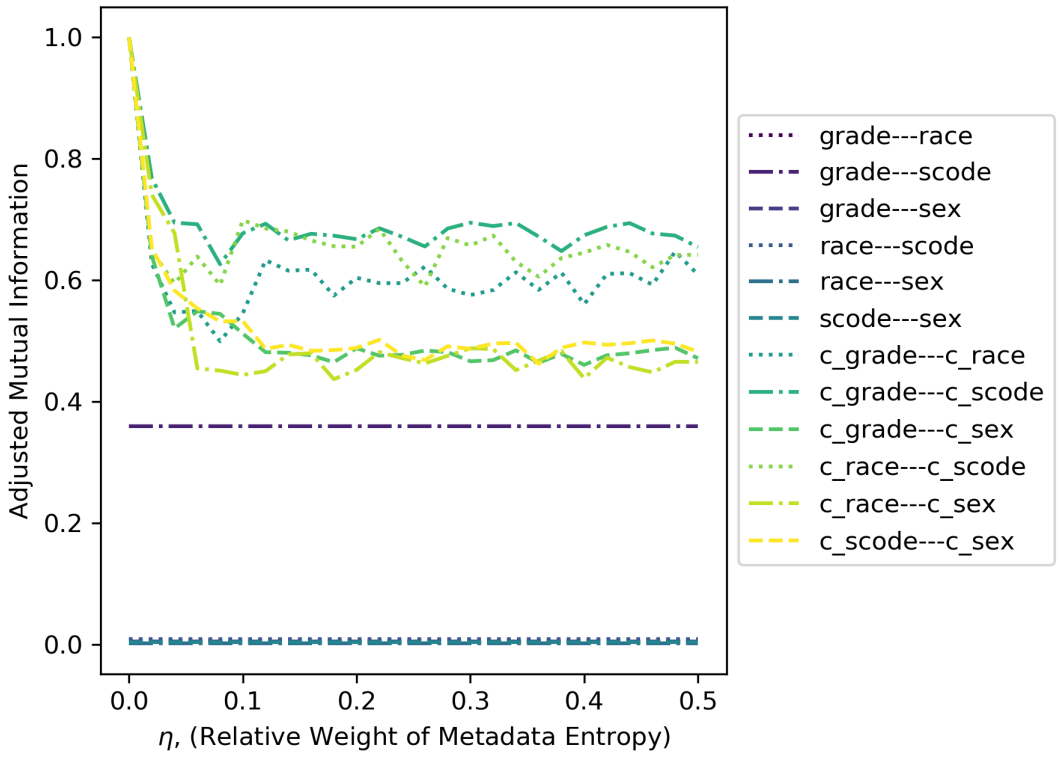

Figure 2.6: Pairwise AMIs of high school social network partitions. For example, "grade" is the partition given by each node's grade, and "c_grade" is the algorithm's returned partition when partitioning with the grade metadata for a given value of η.

with respect to η because the metadata of each node is fixed. Pairs of community detection partitions considering different metadata begin with an AMI close to 1; when $\eta = 0$, the only difference in the returned partitions is due to stochasticity in optimization of the objective function. As η increases, the algorithm returns partitions more aligned with the attribute under consideration, so the pairwise AMIs of these partitions decrease.

One can suppose that the optimal partition at $\eta = 0$ is a point in the space of all possible partitions of the graph. In this interpretation, increasing η for a given metadata type causes the optimal partition to shift in partition space toward partitions more aligned with the particular metadata type. As the optimal partitions for different metadata types undergo such shifts, they diverge in partition space, and as Fig. 2.6 illustrates, their pairwise AMI decreases.

Fig. 2.7 shows another way to understand η. One can consider increasing η as paying a topological entropy price (the sum of the inter-module codebook and intra-module codebook entropies, which is equal to the traditional map equation) for increased AMI with the metadata. The varying shapes of

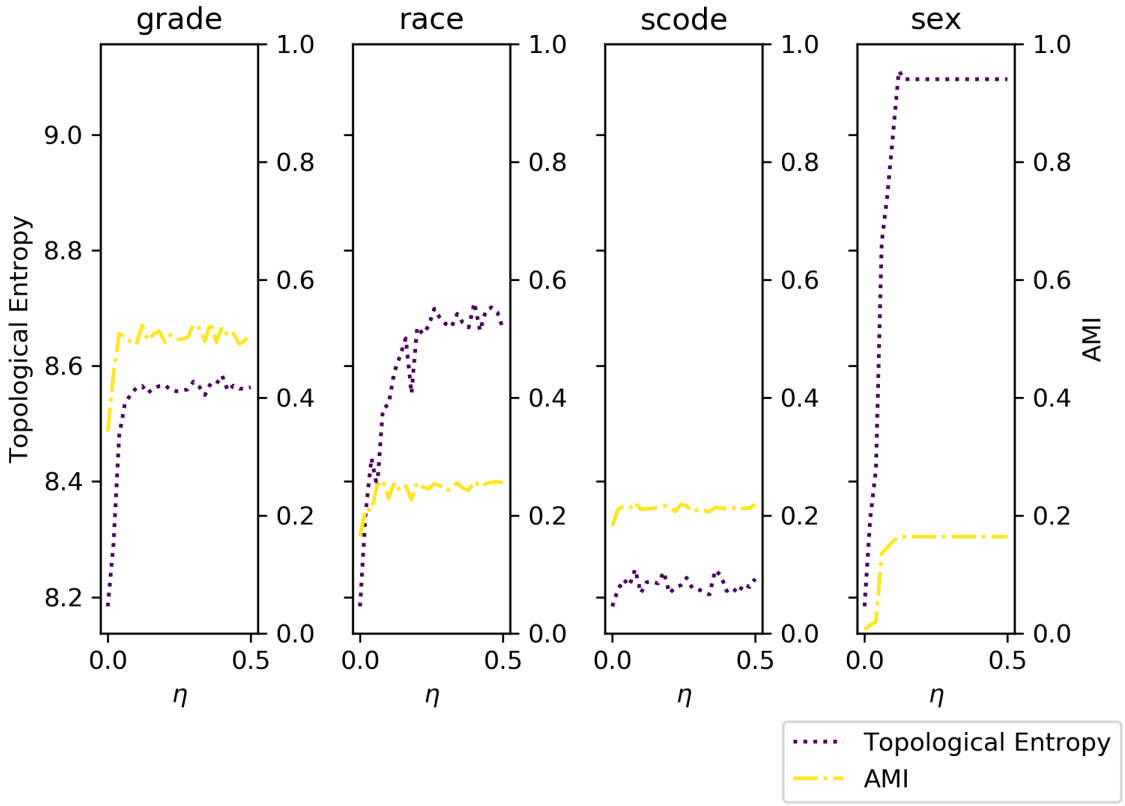

Figure 2.7: Topological entropy and AMI tradeoff when partitioning with metadata in the high school social network. Topological entropy, equal to the traditional map equation, is the sum of the inter-module codebook and intra-module codebook entropies. The AMI is between the node metadata and the algorithm's returned partition.

the curves in Fig. 2.7 show how the price of this tradeoff at a given value of η depends on how node metadata values relate to the network structure. For example, consider the curves corresponding to the school code and sex metadata. For school code metadata, the optimal partition at $\eta = 0$ is already relatively close to meeting the constraint required as $\eta \to \infty$ that each module have just one metadata attribute. One cannot trade topological entropy for much increase in AMI with the metadata because increasing η does not much change the returned partition. For sex metadata, however, the optimal partition at $\eta = 0$ is relatively far from having just one metadata attribute per module. By increasing η, one can pay topological entropy for increased AMI with the metadata as the returned partition shifts toward obeying the constraint imposed as $\eta \to \infty$.

2.5 Summary

We introduced a tuning parameter to the content map equation that explicitly specifies the importance of metadata relative to edge connectivity in assigning communities. We demonstrated on synthetic graphs how focusing on the metadata can overcome the detectability limit when the metadata is well-aligned with the topological community structure and also how focusing on the metadata can put a ceiling on the performance when the metadata is misaligned with the topological community structure. On real-world graphs, we demonstrated how a practitioner might tune the content map equation to "zoom in" and "zoom out" on communities with varying levels of metadata focus.

Our method probes the relationship between community structure and metadata. While we gave the algorithm only one type of metadata attribute at a time, future work might simultaneously incorporate metadata attributes of differing type and relative weighting. It might also be interesting to study how various metadata types relate to various network processes as, for example, different metadata types might relate to the spread of different kinds of information.

CHAPTER 3

A SUPERVISED LEARNING METHOD FOR LINK PREDICTION

3.1 Background

There are many applications of supervised learning on graphs which arise when we have training data and would like to make a prediction on test data. One application is predicting node or link metadata. For example, given a network of student interaction in an online classroom, we might want to predict which students need additional tutoring. Another application is predicting subgraph properties. For example, Dai *et al.* (2016) use molecular graphs to predict which materials in the Harvard Clean Energy Project (Hachmann et al., 2011) have the most solar cell energy efficiently.

Here, we focus on the problem of link prediction: given a graph, can we predict which edges are missing or most likely to form in the future? We ground our analysis in a review of *Decagon*, a state-of-the-art method for link prediction in multimodal graphs with varying node and edge types (Zitnik et al., 2018).

Decagon's development was motivated by polypharmacy side effect prediction. Each year in the United States, 47 million Americans undergo polypharmacy, the simultaneous prescription of at least two drugs (Kantor et al., 2015). Perscribing two drugs at once can cause adverse polypharmacy side effects, side effects that are due to the drug combination and not attributable to either individual drug. Unknown polypharmacy side effects are pervasive because the space of all possible drug combinations is enormous; there are simply too many combinations to test them all. To guide the discovery and prevention of polypharmacy side effects, we would like to predict side effect links in a drug-drug interaction network.

Hypothesizing that drugs' molecular basis — the interaction between proteins and the proteins that drugs target — influences polypharmacy side effects, Zitnik *et al.* develop *Decagon* after constructing a multimodal graph of drugs and proteins. First, they synthesize information about protein-protein interaction from the work of Menche *et al.* (2015), Chatr-Aryamontri *et al.* (2014), Szklarczyk *et al.* (2016a), and Rolland *et al.* (2014), collecting 19,085 proteins with 715,612 protein-protein interactions. All of the interactions were experimentally documented in humans and include various types such as metabolic, enzyme relationships and signaling relationships. Second, they pull information about drugs targeting proteins from the database STITCH (Search Tool for InTeractions of CHemicals) to obtain 18,596 drug-protein edges (Szklarczyk et al., 2016b). Finally, they mine the SIDER (Side Effect Resource), OFFSIDES, and TWOSIDES databases for 4,651,131 drug-drug interactions of 964 unqiue side effect types (Kuhn et al., 2015; Tatonetti et al., 2012).

Although *Decagon* is a general-purpose link prediction algorithm, Zitnik *et al.* develop *Decagon* for particular application to this polypharmacy network because predicting polypharmacy side effects would help reduce the $177 billion that the U.S. spends annually on drug-related problems (Ernst and Grizzle, 2001). *Decagon* falls into the related family of graph representation learning algorithms, which first learn vector space embeddings for a graph before applying traditional machine learning methods (Hamilton et al., 2017b). Previous graph representation learning methods include Laplacian eigenmaps (Belkin and Niyogi, 2002), graph factorization (Ahmed et al., 2013), GraRep (Cao et al., 2015), and HOPE (Ou et al., 2016), which learn node embeddings as a matrix factorization of pairwise node similarities under some scoring function, as well as DeepWalk (Perozzi et al., 2014) and node2vec (Grover and Leskovec, 2016), which learn node embeddings reflecting the structure of a (biased) random walk on the network.

With a graph convolutional encoder like the graph convolutional networks of (Defferrard et al., 2016; Kipf and Welling, 2017) and with a tensor factorization decoder as in (Papalexakis et al., 2017), *Decagon* achieves state-of-the-art performance predicting missing links in the drug polypharmacy network. In the following sections, we detail *Decagon*'s methodology and performance.

3.2 Methodology

Decagon is an end-to-end link prediction method. As input, *Decagon* takes a multimodal graph $G = (\mathcal{V}, \mathcal{E})$ consisting of edges (v_i, r, v_j) from node v_i to node v_j with edge type r. As output, *Decagon* produces $p_r^{ij} = p((v_i, r, v_j) \in \mathcal{E})$, the probability that an edge of type r exists between v_i and v_j, for all pairs of nodes and all types of edges.

To process the graph, *Decagon* proceeds in two stages. First, *Decagon* uses a graph convolutional encoder to produce embeddings $\mathbf{z}_i \in \mathbb{R}^d$, $d \in \mathbb{R}$ for every node in the graph. Then, *Decagon* uses tensor factorization to decode the embeddings into pairwise probabilities for each edge type r. The system is trained end-to-end so that all parameters of the encoder and decoder are optimized for prediction accuracy.

3.2.1 *Decagon*'s Graph Convolutional Encoder

Deep learning methods such as convolutional neural networks have achieved state-of-the-art performance across a wide variety of prediction tasks. With graph-structured data, the trick is defining a suitable neural network architecture. How do we "convolve" over a graph?

Decagon employs a neighborhood-aggregation encoder as used by Hamilton *et al.* (2017a). The neighborhood-aggregation encoder uses the existing graph structure G for its neural network architecture. Beginning with initial node embeddings $\mathbf{h}_i^{(0)}$ for each node in the graph, which can contain input node features or else be a one-hot node indicator, *Decagon* iteratively calculates new node embeddings from the neighboring embedding of each node. Each iteration of the algorithm allows information to travel one step in the graph. By incorporating information from each node's k-hop neighborhood at iteration k, the neural network begins locally and convolves over increasingly global node neighborhoods.

For each node, *Decagon* maintains a hidden state $\mathbf{h}_i^k$ at iteration k that is updated as

$$\mathbf{h}_i^{(k+1)} = \phi\left(\sum_r \sum_{j \in \mathcal{N}_r^i} c_r^{ij} \mathbf{W}_r^{(k)} \mathbf{h}_j^{(k)} + c_r^i \mathbf{h}_i^{(k)}\right). \tag{3.1}$$

The weight matrix $\mathbf{W}_r^{(k)}$ is a learned parameter that conveys specific information for each edge type that is shared globally at iteration k. The normalization constants $c_r^{ij} = 1/\sqrt{|\mathcal{N}_r^i||\mathcal{N}_r^j|}$ and $c_r^i = 1/|\mathcal{N}_r^i|$, where $\mathcal{N}_r^i = \{v_j : (v_i, r, v_j) \in \mathcal{E}\}$, are defined to be symmetric. After K total iterations, the final hidden states are assigned to be the node embeddings: $\mathbf{z}_i = \mathbf{h}_i^{(K)}$.

3.2.2 *Decagon*'s Tensor Factorization Decoder

After the encoder produces a node embedding $\mathbf{z}_i$ for each node, the decoder uses the embeddings to predict the probability $p_r^{ij} = p((v_i, r, v_j) \in \mathcal{E})$ of missing edges. The decoder combines parameters learned about the graph structure for all edge types, such as the node embeddings $\mathbf{z}_i$, with specific parameters learned for each edge type r, leveraging general graph structure to determine a likelihood for particular edge types.

For each potential edge (v_i, r, v_j), the decoder computes a likelihood score

$$g(v_i, r, v_j) = \begin{cases} \mathbf{z}_i^T \mathbf{D}_r \mathbf{R} \mathbf{D}_r \mathbf{z}_i, & \text{when } v_i, v_j \text{ are drugs} \\ \mathbf{z}_i^T \mathbf{M}_r \mathbf{z}_i, & \text{when } v_i, v_j \text{ are proteins} \end{cases} \tag{3.2}$$

which is passed through a sigmoid function to compute the final probability

$$p_r^{ij} = \sigma(g(v_i, r, v_j)). \tag{3.3}$$

When v_i and v_j are both drugs, the decoder combines $\mathbf{R} \in \mathbb{R}^{d \times d}$, a shared parameter across all side effect types, with $\mathbf{D}_r \in \mathbb{R}^{d \times d}$, a diagonal matrix controlling the importance of each embedding dimension for a specific side effect type r. This formulation is a rank-d DEDIDCOM tensor decomposition (Papalexakis et al., 2017), but it varies from prior work by jointly training the node embeddings and tensor factorizaion.

When v_i and v_j are both proteins, the decoder takes a bilinear form, and $\mathbf{M}_r \in \mathbb{R}^{d \times d}$ is a trainable parameter modelling a specific drug-drug interaction of type r.

3.2.3 *Decagon*'s Training Procedure

For its objective function, *Decagon* uses the cross-entropy loss

$$J_r(i,j) = -\log(p_r^{ij}) - E_{n\sim P_r(j)} \log(1 - p_r^{in}). \tag{3.4}$$

This loss maximizes the likelihood of observed edges and a random sampling of observed non-edges. The random sampling of non-edges solves the issue of data imbalance in a sparse graph, and the negative sampling distribution P_r follows that of Mikolov *et al.* (2013). Summing $J_r(i,j)$ over all pairs of nodes and all edge types, *Decagon*'s overall loss is

$$J = \sum_{(v_i,r,v_j)\in\mathcal{E}} J_r(i,j). \tag{3.5}$$

To optimize all model parameters for prediction accuracy, *Decagon* trains end-to-end, backpropagating the gradient of its loss through all parameters of the encoder and decoder.

Zitnik *et al.* train *Decagon* using a number of standard techniques for learning regularization and efficient computation. These techniques, such as mini-batching, dropout, and early stopping, as well the training parameters, such as the learning rate and number of epochs, are detailed in (Zitnik et al., 2018).

3.3 Results

Now that we have given an overview of *Decagon*, we show its performance on the polypharmacy network as measured by Zitnik *et al.* (2018) in comparison to four other link prediction methods:

- **RESCAL tensor decomposition** (Nickel et al., 2011), which decomposes the drug-drug adjacency matrix $\mathbf{X}_r$ for each side effect type r into $\mathbf{X}_r = \mathbf{A}\mathbf{T}_r\mathbf{A}^T$. A side effect between drugs i and j of type r is predicted using $\mathbf{a}_i\mathbf{T}_r\mathbf{a}_j$.

- **DEDICOM tensor decomposition** (Papalexakis et al., 2017), which decomposes the drug-drug adjacency matrix $\mathbf{X}_r$ for each side effect type r into $\mathbf{X}_r = \mathbf{A}\mathbf{U}_r\mathbf{T}\mathbf{U}_r\mathbf{A}^T$. A side effect between drugs i and j of type r is predicted using $\mathbf{a}_i\mathbf{U}_r\mathbf{T}\mathbf{U}_r\mathbf{a}_j$.

Approach	AUROC	AUPRC	AP@50
Decagon	0.872	0.832	0.803
RESCAL tensor decomposition	0.693	0.613	0.476
DEDICOM tensor decomposition	0.705	0.637	0.567
DeepWalk neural embeddings	0.761	0.737	0.658
Concatenated drug features	0.793	0.764	0.712

Table 3.1: Average AUROC, AUPRC, and AP@50 for predicting the 964 different polyharmacy side effects. Reprinted from (Zitnik et al., 2018) under the Creative Commons Attribution BY-NC 4.0 License.

- **DeepWalk** (Perozzi et al., 2014; Zong et al., 2017), which creates node embeddings by using the probability that two nodes co-occur on a random walk of the graph. To predict polypharmacy side effects, the embeddings of drug pairs are concatenated and passed to logistic regression classifiers, one for each side effect type.

- **Concatenated drug features**, which conducts principal component analysis (PCA) on the drug-protein interaction matrix and the individual drug side effect data. The PCA representations for a pair of drugs are concatenated and fed into a gradient boosting trees classifier that outputs polypharmacy side effects.

To determine the parameters for each method, Zitnik *et al.* employ grid search with a validation set. For *Decagon*, they find $K = 2$ iterations to be most effective, using a dimension of 64 hididen units in the first layer and 32 hidden units in the second layer, and they set the dropout rate to 0.1 with a mini-batch size of 512. Zitnik *et al.* measure each method's performance with the area under the receiver-operating characteristic (AUROC), area under the precision-recall curve (AUPRC), and average precision at 50 (AP@50).

As Table 3.1 demonstrates, *Decagon* performs the best across all evaluation metrics. *Decagon* achieves as much as a 0.179 gain in average AUROC compared to alternative methods, and it has a 0.079 gain in average AUROC over the next best method. These results are consistent with those of (Hamilton et al., 2017a,b; Kipf and Welling, 2017) who find that end-to-end models achieve state-of-the-art performance. By comparing *Decagon*, which uses a graph convolutional encoder and a tensor decomposition decoder, against the tensor decomposition methods, we can see how much *Decagon*'s graph convolutional encoder adds to its overall performance.

Best performing side effects	AUPRC	Worst performing side effects	AUPRC
Mumps	0.964	Bleeding	0.679
Carbuncle	0.949	Increased body temp.	0.680
Coccydynia	0.943	Emesis	0.693
Tympanic membrane perfor.	0.941	Renal disorder	0.694
Dyshidrosis	0.938	Leucopenia	0.695
Spondylosis	0.929	Diarrhea	0.705
Schizoaffective disorder	0.919	Icterus	0.707
Breast dysplasia	0.918	Nausea	0.711
Ganglion	0.909	Itch	0.712
Uterine polyp	0.908	Anaemia	0.712

Table 3.2: Highest and lowest *Decagon* prediction accuracies. Reprinted from (Zitnik et al., 2018) under the Creative Commons Attribution BY-NC 4.0 License.

k	Polypharmacy effect r	Drug i	Drug j	Evidence
1	Sarcoma	Pyrimethamine	Aliskiren	(Stage et al., 2015)
4	Breast disorder	Tolcapone	Pyrimethamine	(Bicker et al., 2017)
6	Renal tubular acidosis	Omeprazole	Amoxicillin	(Russo et al., 2016)
8	Muscle inflammation	Atorvastatin	Amlodipine	(Banakh et al., 2017)
9	Breast inflammation	Aliskiren	Tioconazole	(Parving et al., 2012)

Table 3.3: Literature evidence for 5 of *Decagon*'s top 10 predictions. Reprinted from (Zitnik et al., 2018) under the Creative Commons Attribution BY-NC 4.0 License.

The specific polypharmacy side effects for which *Decagon* has the highest and lowest prediction accuracy are listed in Table 3.2. Based on discussions with domain experts and some additional statistical analysis, Zitnik *et al.* argue that *Decagon* is best able to leverage the multimodal graph of drugs and proteins to predict side effects with a strong molecular basis. *Decagon* has the most difficulty, they argue, with common side effects and side effects with a weak molecular basis.

Having trained *Decagon*, Zitnik *et al.* apply it to their entire dataset. For each of *Decagon*'s top 10 predictions, Zitnik *et al.* search the medical literature for evidence of the predicted side effect. Remarkably, Zitnik *et al.* find experimental evidence in the literature for 5 of *Decagon*'s top 10 predictions. Table 3.3 lists these polypharmacy side effects and their experimental evidence in the literature.

Overall, *Decagon* outperforms existing methods predicting polypharmacy side effects in the test dataset, and *Decagon* is able to find new polypharmacy side effects that are experimentally validated by the medical literature.

3.4 Summary

Here, we reviewed *Decagon*, a link prediction algorithm for multimodal graphs (Zitnik et al., 2018). *Decagon* takes a representation learning approach to link prediction, first using a graph convolutional encoder to learn embeddings for each node and then using a tensor decomposition decoder to predict missing edges between pairs of nodes. Some of *Decagon*'s parameters are global, allowing *Decagon* to learn general structure across all different node and edge types in the graph, and some of *Decagon*'s parameters are type-specific, enabling *Decagon* to fine-tune its predictions for individual edge types. *Decagon* is trained end-to-end, learning all its parameters to minimize a final loss function that optimizes for prediction accuracy.

Our review of *Decagon* was grounded in Zitnik *et al.*'s analysis of *Decagon*'s performance on a polypharmacy dataset of drugs and proteins. Polypharmacy side effects are side effects from taking a combination of drugs that are not solely caused by any one drug. On the test dataset, *Decagon* achieves state-of-the-art results, outperforming all other methods by at least 0.079 average AUROC. When applied to the entire dataset, *Decagon* finds new polpyharmacy side effects that have been experimentally documented in the medical literature.

As polypharmacy affects 47 million Americans (Kantor et al., 2015) and drug-related complications cost the United States $177 billion each year (Ernst and Grizzle, 2001), *Decagon* has great potential to improve public health. In the particular domain of polypharmacy, applying *Decagon* to larger, more accurate datasets would facilitate better prediction of new polypharmacy side effects, and *Decagon* could be applied at many stages of the pipeline from drug development to drug prescription in order to prevent polypharmacy side effects. In general, *Decagon*'s success in the domain of polypharmacy suggests that link prediction algorithms could have similar success in other domains. There is still much theoretical work to be done, such as extending graph convolutional methods to link prediction in temporal graphs, and the many opportunities to apply this work will only increase as technology develops.

CHAPTER 4

CONCLUSION

In this work, we studied supervised and unsupervised learning on graphs, focusing on two methods. First, we presented a novel community detection algorithm by introducing a tuning parameter to the content map equation. The unsupervised algorithm is the first that we know of enabling users to "zoom in" and "zoom out" on partitions with varying levels of focus on network metadata. Second, we reviewed *Decagon*, a state-of-the-art algorithm proposed by Zitnik *et al.* (2018) for link prediction. We grounded our analysis of *Decagon* with its application to a polypharmacy drug dataset, showing how *Decagon* is able to predict novel drug side effects with experimental validation in the literature.

As we saw with the content map equation, unsupervised learning methods are highly versatile. Requiring no ground-truth labels, they can look for latent structure in any graph. Unsupervised learning methods often form the basis for exploratory data analysis, and they can uncover useful patterns in a variety of datasets. For example, community detection has yielded correlations with task performance in brain connectivity networks and been used to understand the flow of information in scientific citation networks (Stevens et al., 2012; Rosvall and Bergstrom, 2008).

Evaluating community detection algorithms, however, is difficult because there is no way to determine which behavior is "right." By definition, unsupervised problems provide no ground-truth labels with which to verify methods. Consequently, users of community detection algorithms must take care in justifying the assumptions of their models and providing external validation of their results.

As we saw with *Decagon*, supervised learning methods are becoming quite powerful. Given enough training data, they can learn complex mappings from input data to target labels. The quality of

a supervised learning algorithm is simple to measure: test its accuracy on unseen data. By definition, state-of-the-art prediction accuracy represents well-defined progress.

In many applications, however, we are unable to use supervised learning algorithms. There are many datasets, such as the collection of all webpages on the Internet, for which we do not have "ground truth" clustering labels. And in many other datasets, even if ground truth categories exist, it is prohibitively expensive to obtain them because, for example, they require human evaluation.

Taken together, supervised and unsupervised learning methods are complementary tools. In settings where we lack labels for supervised learning, we can leverage unsupervised learning to generate additional labels. In settings where the quality of unsupervised learning is ill-defined, we can evaluate unsupervised learning results against supervised labels. For example, Weng *et al.* (2014) use the output of an unsupervised method, community detection, as input to a supervised method, a classifier, to yield better performance than either method could obtain alone. For future work, we speculate that there are fruitful opportunities to synthesize unsupervised and supervised learning methods, such as using the output of a metadata community detection algorithm such as the content map equation as input to a link prediction algorithm such as *Decagon*.

BIBLIOGRAPHY

Ahmed, A., Shervashidze, N., Narayanamurthy, S., Josifovski, V., and Smola, A. J. (2013). Distributed large-scale natural graph factorization. In *Proceedings of the 22nd international conference on World Wide Web*, pages 37–48. ACM.

Ahn, Y.-Y., Bagrow, J. P., and Lehmann, S. (2010). Link communities reveal multiscale complexity in networks. *Nature*, 466(7307):761–764.

Banakh, I., Haji, K., Kung, R., Gupta, S., and Tiruvoipati, R. (2017). Severe rhabdomyolysis due to presumed drug interactions between atorvastatin with amlodipine and ticagrelor. *Case reports in critical care*, 2017.

Belkin, M. and Niyogi, P. (2002). Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*, pages 585–591.

Bicker, J., Fortuna, A., Alves, G., Soares-da Silva, P., and Falcão, A. (2017). Elucidation of the impact of p-glycoprotein and breast cancer resistance protein on the brain distribution of catechol-o-methyltransferase inhibitors. *Drug Metabolism and Disposition*, 45(12):1282–1291.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008.

Cao, S., Lu, W., and Xu, Q. (2015). Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 891–900. ACM.

Chatr-Aryamontri, A., Breitkreutz, B.-J., Oughtred, R., Boucher, L., Heinicke, S., Chen, D., Stark, C., Breitkreutz, A., Kolas, N., O'donnell, L., et al. (2014). The biogrid interaction database: 2015 update. *Nucleic acids research*, 43(D1):D470–D478.

Combe, D., Largeron, C., Géry, M., and Egyed-Zsigmond, E. (2015). I-louvain: An attributed graph clustering method. In *Advances in Intelligent Data Analysis XIV*, pages 181–192. Springer.

Dai, H., Dai, B., and Song, L. (2016). Discriminative embeddings of latent variable models for structured data. In *International conference on machine learning*, pages 2702–2711.

Decelle, A., Krzakala, F., Moore, C., and Zdeborov, L. (2011). Inference and phase transitions in the detection of modules in sparse networks. *Physical Review Letters*, 107(6):065701.

Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852.

Edler, D. and Rosvall, M. (2014–2019). The MapEquation software package. `https://mapequation.github.io/infomap/`.

Ernst, F. R. and Grizzle, A. J. (2001). Drug-related morbidity and mortality: updating the cost-of-illness model. *Journal of the American Pharmaceutical Association (1996)*, 41(2):192–199.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3):75–174.

Fortunato, S. and Hric, D. (2016). Community detection in networks: A user guide. *Physics Reports*, 659:1–44.

Ghasemian, A., Hosseinmardi, H., and Clauset, A. (2018). Evaluating overfit and underfit in models of network community structure. *arXiv:1802.10582*.

Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864. ACM.

Hachmann, J., Olivares-Amaya, R., Atahan-Evrenk, S., Amador-Bedolla, C., Sánchez-Carrera, R. S., Gold-Parker, A., Vogt, L., Brockway, A. M., and Aspuru-Guzik, A. (2011). The harvard clean energy project: large-scale computational screening and design of organic photovoltaics on the world community grid. *The Journal of Physical Chemistry Letters*, 2(17):2241–2251.

Hamilton, W., Ying, Z., and Leskovec, J. (2017a). Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017b). Representation learning on graphs: Methods and applications. *IEEE Data Engineering Bulletin*, 40(3):52–74.

He, T., Chan, K. C. C., and Yang, L. (2018). Clustering in networks with multi-modality attributes. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*, pages 401–406.

Hric, D., Peixoto, T. P., and Fortunato, S. (2016). Network Structure, Metadata, and the Prediction of Missing Nodes and Annotations. *Physical Review X*, 6(3):031038.

Kantor, E. D., Rehm, C. D., Haas, J. S., Chan, A. T., and Giovannucci, E. L. (2015). Trends in prescription drug use among adults in the united states from 1999-2012. *Jama*, 314(17):1818–1830.

Kim, Y. and Jeong, H. (2011). Map equation for link communities. *Phys. Rev. E*, 84:026110.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*.

Kuhn, M., Letunic, I., Jensen, L. J., and Bork, P. (2015). The sider database of drugs and side effects. *Nucleic acids research*, 44(D1):D1075–D1079.

Lambiotte, R. and Rosvall, M. (2012). Ranking and clustering of nodes in networks with smart teleportation. *Phys. Rev. E*, 85:056107.

Lazega, E. (2001). *The collegial phenomenon: the social mechanisms of cooperation among peers in a corporate law partnership*. Oxford University Press, Oxford ; New York. OCLC: ocm46846863.

Menche, J., Sharma, A., Kitsak, M., Ghiassian, S. D., Vidal, M., Loscalzo, J., and Barabási, A.-L. (2015). Uncovering disease-disease relationships through the incomplete interactome. *Science*, 347(6224):1257601.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Nadakuditi, R. R. and Newman, M. E. J. (2012). Graph spectra and the detectability of community structure in networks. *Physical Review Letters*, 108(18):188701.

Newman, M. E. J. and Clauset, A. (2016). Structure and inference in annotated networks. *Nature Communications*, 7:11863.

Nickel, M., Tresp, V., and Kriegel, H.-P. (2011). A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 809–816. Omnipress.

Ou, M., Cui, P., Pei, J., Zhang, Z., and Zhu, W. (2016). Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1105–1114. ACM.

Papalexakis, E. E., Faloutsos, C., and Sidiropoulos, N. D. (2017). Tensors for data mining and data fusion: Models, applications, and scalable algorithms. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(2):16.

Parving, H.-H., Brenner, B. M., McMurray, J. J., De Zeeuw, D., Haffner, S. M., Solomon, S. D., Chaturvedi, N., Persson, F., Desai, A. S., Nicolaides, M., et al. (2012). Cardiorenal end points in a trial of aliskiren for type 2 diabetes. *New England Journal of Medicine*, 367(23):2204–2213.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Peel, L., Larremore, D. B., and Clauset, A. (2017). The ground truth about metadata and community detection in networks. *Science Advances*, 3(5):e1602548.

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

Porter, M. A., Onnela, J. P., and Mucha, P. J. (2009). Communities in networks. *Notices of the AMS*, 56(9):1082–1097 & 1164–1166.

Rolland, T., Taşan, M., Charloteaux, B., Pevzner, S. J., Zhong, Q., Sahni, N., Yi, S., Lemmens, I., Fontanillo, C., Mosca, R., et al. (2014). A proteome-scale map of the human interactome network. *Cell*, 159(5):1212–1226.

Rosvall, M. and Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123.

Russo, M. G., Sancho, M. I., Silva, L. M., Baldoni, H. A., Venancio, T., Ellena, J., and Narda, G. E. (2016). Looking for the interactions between omeprazole and amoxicillin in a disordered phase. an experimental and theoretical study. *Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy*, 156:70–77.

Shai, S., Stanley, N., Granell, C., Taylor, D., and Mucha, P. J. (2017). Case studies in network community detection. *arXiv:1705.02305*.

Smith, L. M., Zhu, L., Lerman, K., and Percus, A. G. (2016). Partitioning Networks with Node Attributes by Compressing Information Flow. *ACM Trans. Knowl. Discov. Data*, 11(2):15:1–15:26.

Stage, T. B., Brøsen, K., and Christensen, M. M. H. (2015). A comprehensive review of drug–drug interactions with metformin. *Clinical pharmacokinetics*, 54(8):811–824.

Stanley, N., Bonacci, T., Kwitt, R., Niethammer, M., and Mucha, P. J. (2018a). Stochastic block models with multiple continuous attributes. *arXiv:1803.02726*.

Stanley, N., Niethammer, M., and Mucha, P. J. (2018b). Testing alignment of node attributes with network structure through label propagation. *International Workshop on Mining and Learning with Graphs, MLG18*. arXiv: 1805.07375.

Stevens, A. A., Tappon, S. C., Garg, A., and Fair, D. A. (2012). Functional brain network modularity captures inter-and intra-individual variation in working memory capacity. *PloS one*, 7(1):e30468.

Szklarczyk, D., Morris, J. H., Cook, H., Kuhn, M., Wyder, S., Simonovic, M., Santos, A., Doncheva, N. T., Roth, A., Bork, P., et al. (2016a). The string database in 2017: quality-controlled protein–protein association networks, made broadly accessible. *Nucleic acids research*, page gkw937.

Szklarczyk, D., Santos, A., von Mering, C., Jensen, L. J., Bork, P., and Kuhn, M. (2016b). Stitch 5: augmenting protein–chemical interaction networks with tissue and affinity data. *Nucleic Acids Research*, 44(Database issue):D380.

Tatonetti, N. P., Patrick, P. Y., Daneshjou, R., and Altman, R. B. (2012). Data-driven prediction of drug effects and interactions. *Science translational medicine*, 4(125):125ra31–125ra31.

Taylor, D., Shai, S., Stanley, N., and Mucha, P. J. (2016). Enhanced detectability of community structure in multilayer networks through layer aggregation. *Physical Review Letters*, 116(22):228301.

Vinh, N. X., Epps, J., and Bailey, J. (2010). Information Theoretic Measures for Clusterings Comparison: Variants, Properties, Normalization and Correction for Chance. *J. Mach. Learn. Res.*, 11:2837–2854.

Weng, L., Menczer, F., and Ahn, Y.-Y. (2014). Predicting successful memes using network and community structure. In *Eighth international AAAI conference on weblogs and social media*.

Yang, J., McAuley, J., and Leskovec, J. (2013). Community Detection in Networks with Node Attributes. In *2013 IEEE 13th International Conference on Data Mining*, pages 1151–1156.

Zitnik, M., Agrawal, M., and Leskovec, J. (2018). Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*, 34(13):i457–i466.

Zong, N., Kim, H., Ngo, V., and Harismendy, O. (2017). Deep mining heterogeneous networks of biomedical linked data to predict novel drug–target associations. *Bioinformatics*, 33(15):2337–2344.