Chapter 3
Community Discovery in Multi-Mode Networks

Isaac Jones, Lei Tang and Huan Liu

Abstract As social media becomes more feature-rich and the capability for interactions between users becomes more complex as a result, it may become necessary to expand the models used in data analysis to represent more complex interactions and networks. To that effect, researchers have begun using graphs with different types of vertices or even hyperedges to represent more complex networks. In this chapter, we will explore some of the community detection approaches state-of-the-art research uses to deal with the increasing complexity of social networks, and particularly representing those networks as multi-mode networks (or heterogeneous networks). This chapter will cover the approaches used as well as the graph representations of complex networks. Though the work studied uses social networks as the basis for analysis, the use of multi-mode networks and hyperedges is principled in any analysis task where the complexity of the data calls for multiple types of entities with interactions involving two or more entities in the network.

3.1 Motivation

In November 2010, one of the biggest takedowns of illegal botnets occurred when Dutch authorities systematically dismantled a network of 30 million infected machines.[1] These machines were located across the globe and were responsible for sending out over 30 million spam messages every day. The reason this type of takedown is possible is through community analysis, the detection of community structures. The infected computers in the botnet destroyed by the Dutch authorities forms a community similar, though obviously more nefarious than, the one formed

[1] http://www.zdnet.com/article/dutch-police-take-down-bredolab-botnet/.

I. Jones · H. Liu
Arizona State University, Tempe, USA

L. Tang (✉)
Head of Data Science, Clari Inc., 100 W. Evelyn Ave, Suite 210,
Mountain view, CA 94041, USA
e-mail: leitang@acm.org

G. Paliouras et al. (eds.), *User Community Discovery*, Human–Computer Interaction Series, DOI 10.1007/978-3-319-23835-7_3

by a group of high school classmates or members of a tennis club. By performing some of the same analysis on communication networks that we do on social networks, we can uncover covert networks like the botnets the Dutch authorities did. However, since this 2010 takedown, both social media networks and the botnets that operate online have gotten more complex, to enhance user experience in the former case and to evade detection in the latter case. This may seem like a strange parallel to draw, but both social media networks and communication networks have similarities that make their analysis similar. For example, on Twitter and Facebook the following and friend links (respectively) can be seen as communication pathways between computers in a botnet, and tweets or posts can be seen as the messages that go across these communication pathways.

Since 2010, social media networks have continued to add features. In 2012, Twitter acquired and integrated the features from Vine, a popular video-sharing network. Facebook famously acquired Instagram, and subsequently its feature set, in a billion dollar deal. In addition, Twitter launched Twitter Music in 2013, adding even more features for its users. The addition of these feature layers to their host social media networks makes them more appealing to users, but it also makes the representation of these networks more difficult for community detection and other network analysis. Consider the standard network representation that we might use on a Facebook network, how would we represent the action of commenting on a posted Instagram photo? This newfound complexity is not easily handled by traditional network formulations. One possible way to represent this new data is by adding supplementary graphs, linking users and their photos, users and their comments, comments and the photos they are posted on, and so forth, for as many graphs is necessary. However, this could be inconvenient representation and only grows more complex as more features are added and interactions become richer.

One sensible alternative is to use a *multi-mode* graph. These types of graphs, as we will detail throughout this chapter, are capable of representing an arbitrary number of feature layers in a network and scale quickly and easily to represent interactions between those features. To use the previous example; comments, photos, and users can all be easily represented on one graph without the complexity of maintaining supplementary graphs.

Another challenge beyond is the noisiness of interaction data in communications networks. For instance, as of late 2014, users posted 70 million images to Instagram *every day*, and for some analysis tasks a great portion of those images are bound to be irrelevant. However, a great variety of analysis tasks occur on social media networks, so keeping as many data points as possible is important in case that data becomes relevant in future tasks. This shifting relevance of data makes noise a major concern for researchers. The multi-mode networks are an excellent way to deal with this noise since they remain compatible with the network denoising processes that researchers rely on to winnow down data to only its salient features. These denoising processes rely on the availability of outside information to power their ability to filter out unneeded information. Since multimode networks have that information easily available, in fact built right in to the graph, they can be more effective.

The sheer quantity of noise in the system also makes spurious results inevitable. No system provides perfect accuracy with no false positives, and this provides its own set of challenges for researchers. The risks associated with false positives from data analysis are well documented within the financial auditing community. Dealing with some of the impacts is an unfortunate part of the job in this community. *The Forensic Examiner* featured a six-page article[2] outlining the risks associated with fraud detection specifically and some management strategies for fraud detection experts. Though multi-mode networks do not directly combat the false positive problem, the ability of multi-mode networks to effectively and efficiently factor in more information will assist in making the techniques that utilize them more effective, giving the users better tools for their decision making process.

Another challenge that the enormous size, noisiness, and complexity combine to bring the fore is the volatility of social media network. With billions of users of active users every month, Facebooks social network can hardly be expected to remain static for any appreciable length of time. This has been a continuous challenge for analysts of social media networks [16] and has recently been addressed with various methods that take the changing nature of social networks into account, appropriately classed as *evolutionary* methods. Though we deal only with static network snapshots in this chapter, a number of the methods we look at have either the capability or extensions designed to handle time-varying networks [13]. Key to the evolutionary methods is the ability of multi-mode networks to handle additional information in its network representation, which means that evolutionary methods and multi-mode representations are a natural fit for one another. If the time-dependent structure of the network can be encoded into the multi-mode representation, standard analysis techniques can be used on the networks. This would provide significant potential for analyzing these types of networks, since time information can be easily represented as another mode in the representation.

Multi-mode networks clearly have a significant usefulness when it comes to representing complex social media data and other communication data. The new data demands of increasingly complex social and technical interactions online can be elegantly met by this new network representation that enables and even facilitates analysis. It stands to reason that fields outside of social network analysis can even benefit from using this representation in their techniques. In this chapter, we will discuss three techniques that take advantage of multi-mode networks in their analysis and their results help make the case that this avenue of research is valuable for future work. However, many researchers have considered multi-mode networks in their work, so some terminology may be confusing to readers who aren't familiar with these types of networks already. In the next section, the terminology that will be used in this chapter is introduced and alternative terminology is presented.

[2] http://www.all-about-forensic-science.com/support-files/fraud-detection.pdf.

3.2 Definitions

The feature rich and increasingly complex nature of social networks has caused many researchers to simultaneously develop both methodology and terminology for dealing with these increasingly complex networks. In order to maintain clarity and consistency throughout the chapter, we will start by defining some of the common terms used by researchers, as well as presenting alternative terms with similar meanings used by the research community.

3.2.1 Multi-Mode Networks

In traditional graph theory, an underlying assumption is that all of the vertices in the network being inspected are of the same type. For example, a graph representing a social network might consist of only vertices representing users of that network. Concomitantly, the edges between vertices in this network would represent relationships between the users represented by vertices. Figure 3.1 demonstrates a simple example of such a network.

Networks like those depicted in Fig. 3.1 have received extensive attention both in the existing scientific literature and in previous chapters. As such, the content of this chapter will focus on the more complex networks alluded to above. In this text, and in portions of the existing literature, these networks are referred to as *mulit-mode networks*. In order to better understand this term, it is useful to break it down into its component parts and consider them independently. Here, *network* is used interchangeably with *graph*, and has the same meaning that "graph" does. The other part of the term, *multi-mode*, refers to the computing definition of "mode," a way of operating or using a system. In this case, the system in question is the vertices specified in the network or graph. These vertices adhere to one of a multiplicity of ways of operation in the network.

By way of example, consider the social network discussed above. In the stated formulation, we already represent users of the network as vertices and their explicit relationships as edges. Suppose that the social network also has a messaging feature and we want to represent messages passed from one user to another on the network. We could change the edges to indicate that a message has been sent between two

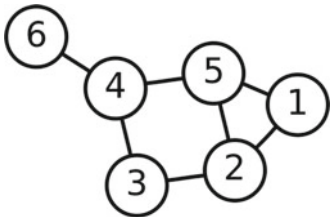

Fig. 3.1 An example of a uni-mode network. Image courtesy of Wikipedia

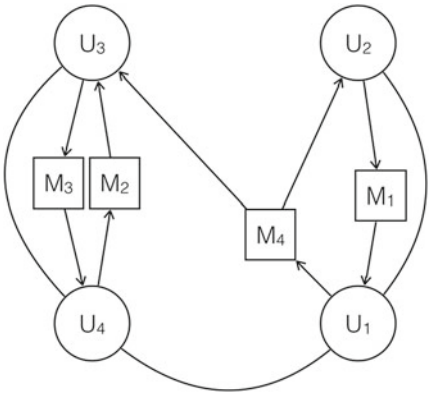

Fig. 3.2 An example social network with message vertices

users, but that has two possible problems. Firstly, it is possible that messages can be sent to more than one person, which would be impossible to capture in the current formulation. There are other edge types that can capture this relationship, which we will discuss in Sect. 3.2.2. Secondly, and possibly more importantly, this erases the information in the network about the relationships between users that may be important for analysis conducted on the network. In order to represent this more complex set of relationships, we can instead add a mode to the existing set of vertices and represent messages sent between users as vertices using this new mode. Then, directed edges going from these new vertices to the users represent the senders and receivers of the messages. And example of a possible network using this formulation is given in Fig. 3.2.

The network in the example above is an example of a graph with two modes, a *bi-mode network* or *2-mode network*. If, instead of the small toy network, we had a very complex network like Facebook[3] where users perform a wide variety of actions like posting images, posting links, posting statuses, commenting on statuses, replying to comments, commenting on fan pages, and so on, representing each of these actions as a mode of the network could result in a very high modality. Thus, for simplicity, any network with more than one mode is simply referred to as a *multi-mode network*.

In addition to the terminology used above, other researchers have other terms they prefer. For example, networks like the one depicted in Fig. 3.2 are referred to in [11] as *heterogeneous* networks. This calls attention to the contrast between the network of Fig. 3.1, which is *homogeneous* because it contains only one type of vertex. In this way, referring to a network as *heterogeneous* is equivalent to referring to it as *multi-mode*. In addition, these types of networks are referred to in [8] as *multi-dimensional networks*, where "dimensional" refers to the increase in matrix dimensions necessary to represent the network in its adjacency matrix form. Note that the same term, *multi-dimensional networks*, is used in [2, 15] to represent a

[3] www.facebook.com.

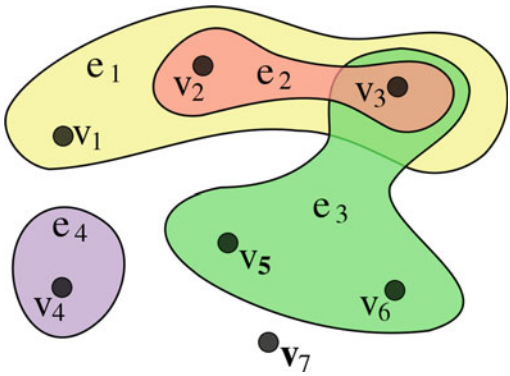

Fig. 3.3 A graph demonstrating hyperedges. Image courtesy of Wikipedia

uni-mode network with multiple types of interactions between vertices of the single mode, which are also referred to as multiplex or multi-relational networks [4].

3.2.2 Hyperedge and Hypergraph

When a network possesses multiple modes, the complexity of interactions between vertices also increases. A *simple edge* represents the interaction between two vertices. This definition of edges can be extended to include more than two vertices, resulting in *hyperedges* as shown in Fig. 3.3. This expansion of simple edge to hyperedge captures more information, but also increases complexity, similar to the way that expanding a network's modality increases complexity. Note that any graph that contains edges of increased complexity like hyperedges is called a *hypergraph*.

In simple uni-mode networks, edges necessarily occur between two vertices of the same type. In multi-mode networks, one edge can involve vertices of one or two modes, or even more than two modes. For example, in [8], the user actions on the Digg[4] network are modeled as a five-mode network with vertices representing users, stories, comments, topics, or keywords. *Hyperedges* are used to represent complex relationships between these objects; for example, a comment action involves vertices from three modes: user, story, and comment.

Since hypergraphs and, particularly, hyperedges are specifically included to capture relationships between vertices, it may be valuable to re-imagine the network of Fig. 3.2 as a hypergraph. Figure 3.4 demonstrates this conversion.

While "hyperedge" and "hypergraph" have their roots in mathematics literature, researchers in computer science often use alternate terminology to refer to these concepts and types of graphs. For example, *multi-relational* is used in [8] to capture the idea that not only do the hyperedges used in the representation capture relations the way that a relational database does, but that there are many different possible types of relations. In addition, *metagraph* is used in [8] to describe a graph that has

[4] www.digg.com.

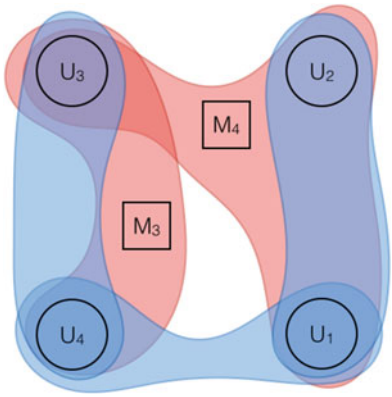

Fig. 3.4 The two-mode graph of Fig. 3.2 converted to a hypergraph. Note that M_1 and M_2 were removed to reduce clutter

both the properties of a hypergraph and a multi-mode graph. Further adding notational complexity, the term *metagraph* is used in [3] to indicate that a graph consists of both standard vertices and aggregate vertices that represent fine-grained communities. Some researchers, however, choose to avoid the complexity of hypergraphs by changing the network representation. For example, in [11], the authors choose to re-formulate the network as a Star Network in order to avoid using hyperedges, which we will discuss in more detail in Sect. 3.4.2.

In this chapter, we will use the term *multi-mode network* to refer to networks with more than one type of vertex and *hyperedges* to refer to edge representations that are more complex than the traditional edges. Accordingly, any graph that contains hyperedges will be referred to as a *hypergraph*.

3.3 Problem Formulation

Community detection on multi-mode networks or hypergraphs requires careful formulation in order to make the problem tractable. While the basis of the formulation remains the same, individual researchers formulate the problem in different ways in order to apply their unique methods to the problem. However, commonalities exist in the formulations. Here, we will discuss the formulation in terms of those commonalities. In Sect. 3.4 we will discuss the specific modifications to the formulation that are necessary for each approach.

3.3.1 Community Detection

Fundamentally, the problem of community detection can be formulated as follows: Given a set of n actors, represented by vertices, $\mathbf{N} = \{n_1, n_2, \ldots, n_n\}$ the task of

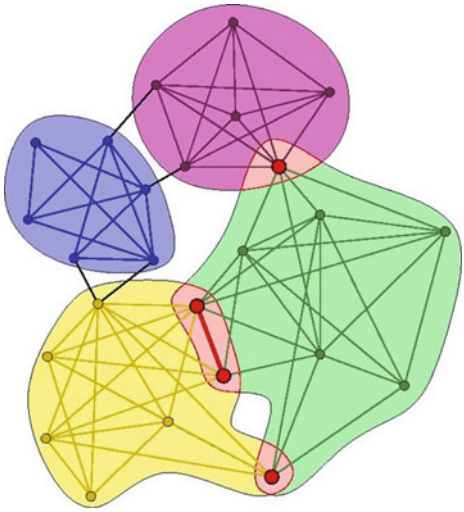

Fig. 3.5 The results of community detection with overlapping communities on a sample network. Image courtesy of mathworks

community detection is to partition the vertices into k groups, called communities, such that $\mathbf{C}_i = \{n_{1_i}, n_{2_i}, \ldots, n_{m_i}\}$. In this case, community C_i is of size m_i, but communities need not be of a uniform size, nor are communities necessarily exclusive. Indeed, an argument can be made (and has, frequently) that non-exclusive communities are more representative of real social networks [12]. Figure 3.5 demonstrates an example of detecting communities on a network with only one type of vertex.

However, this formulation obviously falls short when expanded to multi-mode communities. In multi-mode networks, the set of vertices cannot be as simply partitioned. This is because uni-mode community detection algorithms assume that all vertices in the network are of the same type. Obviously, this assumption is not valid on multi-mode networks. For the example discussed in Sect. 3.2, it would be unsatisfactory if we treat all vertices the same and cluster the vertices representing users and the vertices representing messages into the same cluster. Putting them into one cluster would imply that the vertices are "equivalent." This would allow the clustering algorithm to treat messages and users identically, which could lead to clusters of users including messages, even messages that were not sent by any of the users in the cluster. Thus, the problem formulation must be modified to take into account the mode information for multi-mode networks.

3.3.2 Multi-Mode Communities

As discussed, the variations in vertex modality must be represented, and this additional complexity has cascading effects in the formulation. In this section, we will

cover changes to the basic community detection formulation that must be made to expand to a multi-mode network. In particular, the set of all vertices is defined by:

$$\mathcal{N} = \{\mathbf{N}_t\}_{t=1}^{T}$$

This means that given T types of vertices, the set of all vertices is divided into smaller sets corresponding to the type of vertex. This ensures that each type of vertex is treated differently, unlike the previous formulation. Of course, it would be possible to do community detection on only one of the N_t terms, but as discussed before, this removes valuable information. Instead, each of the T terms in $\mathcal{N}$ is clustered into one of k_t communities. Note that in this case, the number of communities can vary based on the type of the vertex. Thus, what community membership indicates can vary from mode to mode. For example, continuing the example used in Sect. 3.2, the message mode could have a cluster for messages about sports and one for messages about pet care while the user dimensions contains clusterings based on high school graduating class. Certainly, there are members from every graduating class who are interested in sports, and the same for pet lovers.

In order to capture the cross-mode information, we must also capture the relationships between vertices of different modalities. To do this, we define a relationship matrix between vertices. This matrix

$$R^{i,j} \in \mathbb{R}^{|N_{t_i}| \times |N_{t_j}|}$$

represents both the existence and strength of a relationship between elements of modality i and j. Note that in this formulation, i and j can be equal, and this matrix represents the relationships between vertices of the same modality. Typically, 0 is used to indicate that no relationship exists between two given vertices and 1 is used to indicate that such a relationship does exist. However, [0, 1] normalized values have been used by some researchers to indicate the strength of this relationship as well as the presence.

In addition to representing the vertices and their relationships, it is necessary to represent the object of the community detection problem, the communities themselves. In the generic multi-mode community detection problem formulation, we assume that communities are bounded to only one mode of the graph, and so we define *community indicator* or *community membership* matrix as:

$$C^{i,j} \in \{0, 1\}^{|N_{t_i}| \times k_i}.$$

as before, k_i represents the number of communities into which mode i is partitioned. Note that since communities do not transcend modal boundaries, it is not necessary for the set of all C matrices to contain cross-modal relationships, as was required to properly represent the relationships between vertices in the relationship matrix.

In addition to the interactions between vertices in the graph, we also assume that communities in the graph are not totally independent, even across modalities. It would be inappropriate to assume, for example, that the community of messages

about pet care and the community of users interested in pet care artist completely independently. Obviously, these two groups are related, even though they are in different modes of the network. Thus, we define a community interaction matrix

$$A^{i,j} \in R^{k_i \times k_j},$$

Note that this matrix is smaller than the vertex interaction matrix, as the number of communities is much less than the number of vertices. This matrix can also serve as an abstraction of all cross-mode relationships to a smaller feature space.

Note that all the notations above assume that there are no hyperedges in the graph. As we shall see later, these notations can be extended in a consistent way (using tensors, to be discussed later) to handle hyperedges and hypergraphs.

3.4 Methods

In this section, we will discuss three different approaches to performing community detection on multi-mode networks. The methods discussed will be those presented by Tang et al. in [14], Sun et al. in [11], and Lin et al. in [8]. Each of these methods will be referred to by the names the authors gave their systems. These are, respectively: EMMC, NetClus, and Metafac. At the end of this section, we compare all the three methods and discuss their connections.

3.4.1 Evolutionary Multi-Mode Clustering

Presented by Tang et al. in [14], Evolutionary Multi-Mode Clustering (EMMC) is formulated in such a way that it takes community changes over time into account (thus *evolving*). However, this does not affect its ability to detect communities in multi-mode networks. While evolutionary clustering is an emerging field of community detection research and has its own challenges and applications, it is not within scope for this chapter, so the evolutionary portion of this work will be disregarded.

The EMMC method starts out with an assumption that the relationship matrix R captures interactions of the communities of interest in the graph. Mathematically:

$$R^{i,j} \approx C^i A^{i,j} (C^j)'$$

Here, $A^{i,j}$ is the confounding factor that links the community memberships in mode i and j, which is the community interaction matrix discussed in the previous section. Consequently, it is reasonable to attempt to estimate the community membership of two modes of the network as:

$$\min \ ||R^{i,j} - C^i A^{i,j} (C^j)'||_F^2 \tag{3.1}$$

$$s.t.\ \ C^i \in \{0,1\}^{|N_{t_i}| \times k_i} \quad \sum_{k=1}^{k_i} C_k^i = 1 \tag{3.2}$$

$$C^j \in \{0,1\}^{|N_{t_j}| \times k_j} \quad \sum_{k=1}^{k_j} C_k^j = 1 \tag{3.3}$$

The objective function in Eq. (3.1) attempts to minimize the Frobenius Norm of two matrices. Combining this objective for all cross-mode interactions, it follows that:

$$\min \sum_i \sum_j \ w_{i,j} ||R^{i,j} - C^i A^{i,j} (C^j)'||_F^2 \tag{3.4}$$

$$s.t.\ \ C^i \in \{0,1\}^{|N_{t_i}| \times k_i}, \quad \sum_{k=1}^{k_i} C_k^i = 1 \quad \text{for all } i = 1, 2, \ldots, T \tag{3.5}$$

where $w_{i,j}$ is the weight assigned to the interaction between modes i and j. Summing up the results over all pairs of dimensions ensures that the relationships between all pairs of modes are taken into account. Since there is no constraint that $i \neq j$, communities within the same mode are also taken into account.

However, due to the discrete nature of constraints in Eq. (3.5), the minimization problem is NP-Hard. In order to alleviate this issue, the authors use a well-studied technique in the spectral clustering literature [17]. This technique is to relax the discrete constraints into continuous constraints. By using continuous constraints, we can think of the community indication matrix as indicating a "level of membership" for each community. This also has the effect of transforming the single-community solution proposed initially to an overlapping community solution. Since the community indication matrix can now have more than one non-zero value, we can say that a vertex belongs primarily to the community with which it has the highest value, but partially to all of the other communities with which it has a non-zero value. This relaxation transforms the constraints in Eq. (3.5) into:

$$(C^i)' C^i = I_{k_i}. \tag{3.6}$$

In English, this mean that the community indicator matrix is column orthogonal, which yields the following final problem formulation:

$$\min \sum_i \sum_j \ w^{i,j} \ ||R^{i,j} - C^i A^{i,j} (C^j)'||_F^2 \tag{3.7}$$

$$s.t.\ \ (C^i)' C^i = I_{k_i} \tag{3.8}$$

There is no analytical solution to the problem above because of the simultaneous unknowns C^i, C^j, and $A^{i,j}$. However, alternating optimization can be adapted to

solve it. That is, we can compute one variable while fixing all other variables. When fixing C^i and C^j, the optimal $A^{i,j}$ is as follows:

$$A^{i,j} = (C^i)^T R^{i,j} C^j \tag{3.9}$$

It can be shown that the optimal C^i, given all other variables, is the left singular vector of a matrix P that consists of several sub-matrices concated column-wise:

$$P^i = \left[\left\{ \sqrt{w^{i,j}} R^{i,j} C^j \right\} \right] \tag{3.10}$$

Hence, EMMC can be solved iteratively. In each iteration, we cycle through all modes and then update C^i as the left singular vectors of the matrix P^i.

3.4.2 Net-Clustering

Sun et al. describe in [11] a method for performing multi-mode community detection based on transforming a traditional multi-mode network into a type of network called a *star network*. This modification to the network results in some changes to the problem formulation we presented in Sect. 3.3, as well as requiring discussion of the Star Network modification for clarity.

In order to understand the NetClus formulation, it is important to first discuss the network representation the authors use for the multi-mode network. Consider the original formulation of the network:

$$G = \langle \mathcal{N}, E \rangle$$
$$\text{where } \mathcal{N} = \{\mathbf{N}_t\}_{t=1}^{T}$$

In the formulation used in [11], an additional element is added to the graph definition, a set of weights on the set of edges E, these weights (denoted W for the set and W_{n_i,n_j} for the weight of an individual edge) correspond to the real-values entries of the adjacency matrix R previously mentioned. In addition, the authors of NetClus add another constraint to the network, the *star network* constraint.

The star network constraint imposes a limitation on the connectivity of edges in the network. Normally, an edge in the network is represented by $e = \langle n_i, n_j \rangle$ where the vertices n_i and n_j can be from any modality in T. However, the star network constraint designates a particular member of T ($t = 1$ for simplicity) as the *target type* and forces all edges to have exactly one endpoint in the target type. That is, the set of all edges E has an additional constraint that:

$$\forall e \in E, e = \langle n_i, n_j \rangle, n_i \in N_1, n_j \in N_t (t \neq 1) \tag{3.11}$$

Note that this also imposes the constraint that vertices in the target type may not be connected to one another. For notational purposes, modes of the network other than the first mode are referred to as *attribute types*. Because the star network scheme places such a high emphasis on the target type, using this scheme facilitates the clustering in the mode of the target type, thus the term "target." However, forcing edges to have one end in the target type reduces the ability of a star network formulation to represent real-world graphs. The advantage of using a star network formulation in this context is that the removal of edges between vertices of the target types forces a communities detected in the target type to have attribute vertices linking the vertices of the target type. Based on the attribute vertices used as linkages, this forces a community to have some meaning to human interpreters, serving as both a explanation for the community's existence and a sanity check on that community's existence. These modification to the formulation of the problem power the strengths of the NetClus algorithm.

In particular, the authors define a *net cluster* as a cluster that consists of a target vertex and its highly relevant attribute vertices. Though the authors model the domain they study as a multi-mode star network, it is perhaps more intuitive to interpret the cluster definition and algorithm by thinking of target vertices as objects and attribute vertices as the object's different attributes. Consequently, a cluster is mainly composed of target vertices and their most frequently considered attributes.

Unlike EMMC, NetClus does not minimize an matrix approximation error function to obtain its results. Instead, the algorithm adopts a k-means-like method to compute the most likely assignments of target type vertices to clusters. The NetClus algorithm can be described by the following steps, according to the authors in [11]:

1. Generate initial partitions for target objects and induce initial net-clusters from the original networks according to these partitions, i.e., $\{C_i^0\}_{i=1}^k$;
2. For each cluster, compute out the conditional probability that one attribute vertex is associated with the cluster. i.e., $\{P(x|C_i^t)\}_{i=1}^k$;
3. Calculate the posterior probabilities for each target object $(p(C_i^t|x))$ and then adjust their cluster assignment according to the new measure defined by the posterior probabilities to each cluster.
4. Repeat Steps 2 and 3 until the cluster does not change significantly, i.e., $\{C_i^*\}_{i=1}^k = \{C_i^t\}_{i=1}^k = \{C_i^{t-1}\}_{i=1}^k$.
5. Calculate the posterior probabilities for each attribute object $(p(C_k^*|x))$ in each net-cluster.

Step 2 can be thought of as finding a ranking on the target vertices to best describe the interactions observed in the cluster. This distribution is similar to using "sufficient statistics" to describe the cluster, which is comparable to the cluster centroid in classical k-means clustering algorithm. Step 3 then updates the cluster assignment for each attribute vertex.

As a general approach to the problem, NetClus is an exemplar of methods based on iterative improvement of a model that mathematically describes the data. In this case, the authors of NetClus chose to use a probabilistic model that iteratively improves clusters based on the probability that a particular instance of the target type with its

linked attribute type objects is generated by the clusters the algorithm identifies as being strongly related.

3.4.3 MetaGraph Factorization

Presented by Lin et al. in [8], the final method, MetaFac extends conventional multi-mode networks to include hyperedges as discussed in Sect. 3.2. Unlike EMMC, where all edges are interactions between exactly two vertices and possibly of different modes, MetaFac tackles the cases when multiple modes interact simultaneously. In order to handle community discovery in these hypergraphs, we must extend the traditional matrix representation to *tensors*, a form of higher-order matrices, to represent hyperedges. Tensors are a useful mathematical construct for representing hypergraphs due to their flexibility and the properties of the construction. A brief overview of tensor mathematics is given in [8] and further information can be found in [1]. Though tensor mathematics is not within scope here, some coverage will be necessary in the course of discussing the problem formulation.

We start with simple case of conventional edges representing two-way interactions. Let x_{ij} represent the interaction of two entities (possibly from different modes), k denote a community, $p_{k\to i}$ indicate how likely an interaction in the kth community involves entity i, and p_k be the probability of any interaction in the kth community. Given these things, the probability of interaction can be approximated by:

$$x_{ij} \approx \sum_k p_k \cdot p_{k\to i} \cdot p_{k\to j} \tag{3.12}$$

A 3-way interaction is a simple extension:

$$x_{i_1 i_2 i_3} \approx \sum_k p_k \cdot p_{k\to i_1} \cdot p_{k\to i_2} \cdot p_{k\to i_3} \tag{3.13}$$

Hence, a set of interactions among three modes can be rewritten as

$$\mathscr{X} \approx \sum_k p_k \cdot u_k^{(1)} \circ u_k^{(2)} \circ u_k^{(3)} = [\mathbf{z}] \prod_{m=1}^{3} \times_m \mathbf{U}^{(m)} \tag{3.14}$$

Here $\mathscr{X}$ is the data tensor representing all interaction involving three modes and $\mathbf{Z}$ is the core tensor.

Using tensors, as mentioned above, the authors of [8] formulate MetaFac as an optimization problem. The known inputs to the problem solution are the metagraph[5] $G = \langle V, E \rangle$. Unlike the original definition of a graph, E is represented by a tensor

[5]Recall from Sect. 3.2, that a metagraph is a multi-mode graph by another name.

that involves multiple modes for each hyperedge in the original multi-mode graph. These edges are represented as $\mathscr{X}^{(i)}$, where i is a unique value for each hyperedge. The authors of MetaFac chose KL-divergence [7] (denoted $D(\cdot||\cdot)$) to determine the quality of their estimation. Since the KL-divergence is only defined on a single relation (one instance of $\mathscr{X}^{i}$), the contribution to the total divergence from each relation must be summed up as follows:

$$\min \sum_{r \in E} D(\mathscr{X}^{(r)} || [\mathbf{z}] \cdot \prod_{m:v^{(m)} \in e^{(r)}} \mathbf{U}^{(m)} \tag{3.15}$$

$$s.t. \mathbf{U}^{(q)}_{ik} = 1 \; \forall q \forall k \tag{3.16}$$

Note that in Eq. (3.15), $v^{(m)} \in e^{(r)}$ is used to indicate that $v^{(m)}$ is one of the vertices involved in hyperedge r. By minimizing this cost over the two free variables, **[z]** and $\mathbf{U}^*$, we can find community interaction core tensor, represented by **[z]**, and a set of community memberships, represented by $\mathbf{U}^*$, that reconstruct the observed data as accurately as possible. This is subject to some regularizing constraints as specified in Eq. (3.16). to ensure that $\mathbf{U}^{(q)}$ is a representation of interaction probabilities.

Optimizing the MetaFac objective function is non-trivial and we encourage reader to refer to the source paper for algorithm details. Generally speaking, the authors propose to perform an iterative process. At each iteration, the author compute the optimal [**z**] and $\mathbf{U}^*$ alternatively.

3.4.4 Discussions

In the previous subsections, we have briefly reviewed three different approaches to find communities for multi-mode networks. Below we summarize the data sets studied for each method.

Data and Comparison For EMMC, two network data sets are selected; the Enron email corpus, made public in the wake of the October 2001 scandal, and Digital Bibliography & Library Project (DBLP) data. For the Enron data, the authors constructed a three-mode network. The three modes used are users, messages (e-mails), and words. Users are linked to both the messages that they send and those that they receive. Messages and words are in turn linked together by usage in a particular message. Figure 3.6 show a visual representation of the cross-mode linkages in the data set. The DBLP data is modeled as a network with four modes: papers, authors, terms (words in the title), and venues (conferences or journals). The cross-modal links used for this dataset are the obvious ones, and are depicted in Fig. 3.7.

The authors of NetClus use a similar, but not identical, DBLP data set, setting paper as the target mode and venue, author, and term the other attribute modes.

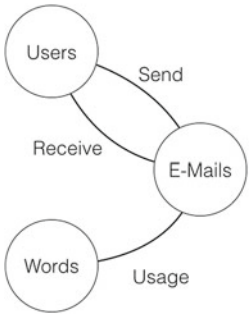

Fig. 3.6 A visual representation of the cross-mode links in the Enron E-mail corpus

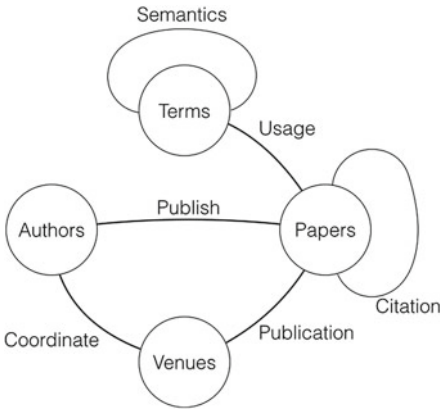

Fig. 3.7 A visual representation of the cross-mode links in the DBLP corpus

Table 3.1 Relations used in MetaFac performed on the Digg datasets and the modes they connect

Relation	Modes
Content	Story, Keyword, Topic
Contact	User, User
Submit	User, Story
Digg	User, Story
Comment	User, Story, Comment
Reply	User, Comment

The authors of MetaFac model the activities on Digg[6] using hyperedges. The data set contains five modes: users, stories, comments, keywords, and topics. In addition, it contains hyperedges connecting these modes. Table 3.1 describes the different hyperedges and the modes they connect. Note that in this dataset, Submit and Digg connect the same modes.

At first glance, the three methods look very different, because each method proposes to handle different types of multi-mode networks. In particular, EMMC aims to handle general multi-mode networks with only two-way interactions. MetaFac

[6] www.digg.com.

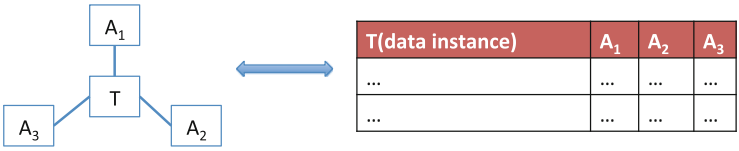

Fig. 3.8 Conversion between star network and data-attribute table

can be thought of as an extension of EMMC with a modification to handle care of multi-way interactions represented by hyperedges, though different loss functions are used for approximation. NetClus, on the contrary, transforms a multi-mode network into a star network. Note that not all networks can be represented by this star network schema. Taking advantage of this schema, a k-means like algorithm can be adopted to compute the communities of target mode vertices. In terms of algorithms, all three methods are iterative, with the clustering results of one mode updated based on the clustering or Interaction probability of other modes.

Connections A multi-mode network representation blurs the conventional definition of *attributes*, *vertices*, and *relations*. For example, NetClus studies multi-mode networks with one target mode and others being attribute modes. Since vertices in attribute modes interact only with those of the target mode, we can think of them as attributes of the target mode vertices to which they are connected. Equivalently, each attribute mode represents one feature, and vertices in that mode become different attribute values, as shown in Fig. 3.8. This makes the NetClus algorithm looks very much like a k-means clustering algorithm handling data associated with different attributes.

On the other hand, hyperedges complicate the community detection problem in multi-mode networks. However, hyperedges can be "flattened," and thus reduced to normal edges, by adding more modes to the network. We can create one additional mode for each hyperedge relation, then each hyperedge becomes one vertex in that mode. Obviously, all the modes involved in the original hyperedge will be connected to this new mode. Figure 3.9 demonstrates such a change. Essentially, a hyperedge relation involving m modes is converted into m two-way interactions between the m modes and one newly created hyperedge mode. Interestingly, such a change is

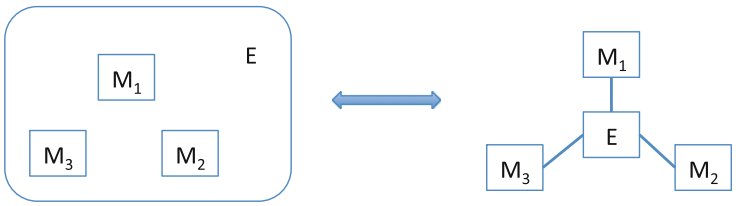

Fig. 3.9 Conversion between hyperedge and flattened multi-mode network

exactly a *star network* schema. However, when multiple hyperedges are observed in the original network, then the modes after conversion will interdependent, returning to the general multi-mode network EMMC deals with.

3.5 Extensions

As alluded to in Sect. 3.4, community detection efforts have multiple extensions above and beyond simply detecting communities. In both [11] and [14], the basic community detection problem is extended into an evolving communities problem. In this section, the community detection extensions of Community Evolution, Link Prediction, and Ranking will be introduced and briefly discussed.

3.5.1 Community Evolution

The first extension, *community evolution*, deals with the time-varying nature of social networks. Part of the nature of social networks is that they are fundamentally dynamic constructs. Links between users are constantly formed and removed, and with these links community structures change; sometimes growing, sometimes shrinking. In some cases, communities merge together and become one larger community. The work described in [14] and [11] contains a discussion of evolving communities as well as simple modifications to the models already presented that enable analysis of community evolution. As demonstrated by [5], evolutionary clustering continues to be a popular extension of community detection as social networks become more mature and more time-varying data is available.

3.5.2 Link Prediction

Communities detected by community detection algorithms inherently represent areas of denser connectivity in the underlying graph. In standard community detection, *modularity*, a measure of the density of connections inside the community compared with those outside, is commonly used as an indicator of community quality [9]. It is reasonable, then, to assume that areas of the graph that have high density of connections will also have a high number of new connections. Thus, it is reasonable to assume that detected communities will indicate locations where new links are likely to form. MetaFac, the method discussed in Sect. 3.4.3 was used by the authors of [8] to make predictions on new links between entities. The results of this prediction demonstrate the potential of community detection algorithms to supplement link prediction algorithms.

3.5.3 Ranking

Lastly, in the real world, each community tends to have a single individual or small group that are highly influential. The problem of finding influential users or vertices in network data is a longstanding problem. This has lead to the development of algorithms like PageRank [10] and its precursor, HITS [6]. Finding influential users given the network is a well-studied problem, but like the work in Link Prediction, ranking problems can benefit from exploiting community structure. The NetClus work described in Sect. 3.4.2 also contains a ranking component to find highly ranked conferences in the Data Mining area [11]. The rankings they find from the DBLP data set match their evaluation of the conferences present in the data set.

3.6 Summary

Community detection [13], and by extension multi-mode community detection, is a highly dynamic, fast-moving field. Multi-mode community detection, in particular, has great potential to provide insight into networks that are becoming increasingly complex with the evolution of social media. As social networks become increasingly expressive and allow their users to conduct more and more of their daily business online, the modality of the network and size of the possible relationship set will have to increase to compensate for this. Current trends in social media support the idea that social networks will become increasingly complex. Facebook[7] continues to add more and more ways for users to interact with one another. Twitter[8] recently debuted the Vine[9] video-sharing service as a way to integrate another form of media, in this case video, into the Twitter social network. Google has made great (albeit controversial) steps by integrating their services together and thus increasingly the number of ways that users of Google's services can interact with one another. Recently, Google opted to integrate Google Plus services with their widely popular YouTube service, requiring users to use their Google Plus identity to comment on YouTube videos. While this was unpopular with YouTube's users,[10] it is easy to see from a community detection perspective how this change massively increased the amount of available data, both in size and complexity.

As the data sets for detecting communities in multi-mode communities become larger and larger, increasingly sophisticated algorithms are needed to draw meaningful conclusions from that data. The various motivational reasons discussed in Sect. 3.1 drive both commercial interests and more academically inclined researchers to strive for better and better community results. The various cross-disciplinary applications

[7] www.facebook.com.

[8] www.twitter.com.

[9] https://vine.co/.

[10] www.forbes.com/sites/insertcoin/2013/11/09/google-plus-creates-uproar-over-forced-youtube-integration/.

discussed briefly in Sect. 3.5 make community detection a widely followed research area, as researchers from a wide variety of disciplines incorporate the latest results from community detection into their work to improve performance.

Acknowledgments We would like the thank Dr. Papadopoulos and the other editors and reviewers for their helpful comments about the content of this chapter, as well as the members of the DMML lab at ASU for the same.

References

1. Bader BW, Kolda TG (2006) Algorithm 862: Matlab tensor classes for fast algorithm prototyping. ACM Trans Math Softw (TOMS) 32(4):635–653
2. Berlingerio M, Coscia M, Giannotti F, Monreale A, Pedreschi D (2013) Multidimensional networks: foundations of structural analysis. World wide Web 16(5–6):567–593
3. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech: theory Exp 2008(10): P10008
4. Cai D, Shao Z, He X, Yan X, Han J (2005) Community mining from multi-relational networks. In: Knowledge discovery in databases: PKDD 2005. Springer, pp 445–452
5. Gupta M, Aggarwal C, Han J, Sun Y (2010) Evolutionary clustering and analysis of heterogeneous information networks. Technical report, IBM research report
6. Kleinberg JM (1999) Hubs, authorities, and communities. ACM Comput Surv (CSUR) 31(4es):5
7. Kullback S, Leibler RA (1951) On information and sufficiency. Ann Math Stat 22(1):79–86
8. Lin YR, Sun J, Castro P, Konuru R, Sundaram H, Kelliher A (2009) MetaFac: community discovery via relational hypergraph factorization. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 527–536
9. Newman ME (2006) Modularity and community structure in networks. Proc Natl Acad Sci 103(23):8577–8582
10. Page L, Brin S, Motwani R, Winograd T (1999) The PageRank citation ranking: bringing order to the web
11. Sun Y, Yu Y, Han J (2009) Ranking-based clustering of heterogeneous information networks with star network schema. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 797–806
12. Tang L, Liu H (2009) Relational learning via latent social dimensions. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 817–826
13. Tang L, Liu H (2010) Community detection and mining in social media. Synth Lect Data Min Knowl Discov 2(1):1–137
14. Tang L, Liu H, Zhang J, Nazeri Z (2008) Community evolution in dynamic multi-mode networks. In: Proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 677–685
15. Tang L, Wang X, Liu H (2009) Uncovering groups via heterogeneous interaction analysis. In: ninth IEEE international conference on data mining, ICDM'09. IEEE, pp 503–512
16. Zafarani R, Abbasi MA, Liu H (2014) Social media mining: an introduction. Cambridge University Press, Cambridge
17. Zha H, He X, Ding C, Gu M, Simon HD (2001) Spectral relaxation for k-means clustering. NIPS 1:1057–1064