FacetNet: A Framework for Analyzing Communities and Their Evolutions in Dynamic Networks

Yu-Ru Lin[1] Yun Chi[2] Shenghuo Zhu[2] Hari Sundaram[1] Belle L. Tseng[3]
[1]Arts, Media and Engineering Program, Arizona State University, Tempe, AZ 85281, USA
[2]NEC Laboratories America, 10080 N. Wolfe Rd, SW3-350, Cupertino, CA 95014, USA
[3]YAHOO! Inc., 2821 Mission College Blvd, Santa Clara, CA 95054 USA
[1]{yu-ru.lin,hari.sundaram}@asu.edu, [2]{ychi,zsh}@sv.nec-labs.com, [3]belle@yahoo-inc.com

ABSTRACT

We discover communities from social network data, and analyze the community evolution. These communities are inherent characteristics of human interaction in online social networks, as well as paper citation networks. Also, communities may evolve over time, due to changes to individuals' roles and social status in the network as well as changes to individuals' research interests. We present an innovative algorithm that deviates from the traditional two-step approach to analyze community evolutions. In the traditional approach, communities are first detected for each time slice, and then compared to determine correspondences. We argue that this approach is inappropriate in applications with noisy data. In this paper, we propose *FacetNet* for analyzing communities and their evolutions through a robust *unified* process. In this novel framework, communities not only generate evolutions, they also are regularized by the temporal smoothness of evolutions. As a result, this framework will discover communities that jointly maximize the fit to the observed data and the temporal evolution. Our approach relies on formulating the problem in terms of non-negative matrix factorization, where communities and their evolutions are factorized in a unified way. Then we develop an iterative algorithm, with proven low time complexity, which is guaranteed to converge to an optimal solution. We perform extensive experimental studies, on both synthetic datasets and real datasets, to demonstrate that our method discovers meaningful communities and provides additional insights not directly obtainable from traditional methods.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data mining*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*; J.4 [**Computer Applications**]: Social and Behavioral Sciences—*Economics*

General Terms

Algorithms, Experimentation, Measurement, Theory

Keywords

Community, Evolution, Soft Membership, Non-negative Matrix Factorization, Community Net, Evolution Net

1. INTRODUCTION

Data from many social network datasets, including paper co-authorship networks and the blogosphere, is a graph where nodes represent individuals (e.g., club members, authors, and bloggers) and edges represent the relationship and interactions among individuals (e.g., interactions in a club, co-authorship, and hyperlinks in blogs). In such social networks, individuals form communities by building relationships and interactions with each other. The analysis of these communities (membership, structure and temporal dynamics) is an important research issue.

Traditional analysis of social networks treats the network as as a *static* graph, where the static graph is either derived from aggregation of data over all time or taken as a snapshot of data at a particular time. These studies range from well-established social network analysis [22] to recent successful applications such as HITS and PageRank [7, 15]. However, this research omits one important feature of communities in networked data— the temporal evolution of communities. By ignoring community evolution, prior works have overlooked a key aspect of online communities.

More recently, there has been a growing body of work on the analysis of communities and their temporal evolution in *dynamic networks* [1, 8, 9, 10, 11, 16, 19, 21]. However, a common weakness in these studies, as we will discuss in detail in related work, is that communities and their evolutions are studied separately—usually community structures are independently extracted at consecutive timesteps and then in retrospect, evolutionary characteristics are introduced to explain the difference between these community structures over time. Such a two-stage approach may make sense when the community structure is unambiguous (e.g., when the community affiliation is available). However, more often than not, data from real-world networks are ambiguous and subject to noise. Under such scenarios, if an algorithm extracts community structure for each timestep independently of other timesteps, it often results in community structures with high temporal variation. Consequently, undesirable evolutionary characteristics may have to be introduced in order to explain the high variation in the community structures. Therefore, we argue that a more appropriate approach is to analyze communities and their evolutions in a unified framework where the community structure provides evidence about community evolutions and at the same time, the evolutionary history offers hints on what community structure is more appropriate. For example, a community structure that introduces dramatic evolutions in a very short period of time is less desirable.

Another common problematic issue in current community analysis techniques is that an individual is usually assigned to only one community at a time. On the contrary, an individual may be engaged in multiple communities at the same time. For example, a blogger who is a dance guru may also be an amateur photographer at the same time. Because of this, an individual who usually participates in multiple communities should be assigned to multiple communities at the same time. Therefore, instead of a hard community partition, we argue that a *soft* community membership is more informative, as it provides more details about how an individual participates in each of the communities.

In this paper, we propose a systematic framework for analyzing communities and their evolutions in dynamic networks, and we term our framework *FacetNet*[1]. Our main contributions are threefold:

1. We introduce the *FacetNet* framework to analyze communities and their evolutions in a unified process. In our framework, the community structure at a given timestep t is determined both by the networked data at t and by the historic community evolution patterns. As a result, the discovered communities and their evolutions are more robust to noise and more reasonable (e.g., dramatic change in a short time is unlikely).

2. We extend the soft clustering algorithm by Yu et al. [24] from static graphs to dynamic networks. In contrast to a hard community partition, in our framework an individual can participate in multiple communities at the same time and with different participation levels. Similarly, an observed relationship is generated due to a combined effect from various communities. Based on the soft community membership, we further define two novel concepts—*Community Net* and *Evolution Net*—to represent community structures and their evolutions, respectively.

3. We provide an iterative algorithm that is guaranteed to converge to (local) optimal solutions to the proposed formulation. We prove the correctness and convergence of our algorithm and show that this algorithm has low time complexity. We also provide principled solutions to some practical issues, such as how to determine the number of communities and how to handle adding and removing of individuals in a dynamic network.

We use synthetic and real datasets (including a blog dataset and a paper co-authorship dataset) to demonstrate that compared to traditional methods, our framework provides more reasonable results on communities and their evolutions. We also show that our framework is able to discover interesting insights in dynamic networks that are not directly obtainable from existing methods.

The rest of the paper is organized at follows. First, we discuss related work. In Section 2, we describe our basic framework in detail. In Section 3, we introduce extensions of our framework to handle some practical issues. In Section 4, we provide experimental studies. Finally in Section 5, we give the conclusion.

[1]*FacetNet* stands for "a Framework for Analyzing Communities and EvoluTions in dynamic NETworks".

Related Work

Community formation has been extensively studied in various research areas such as social network analysis, Web community analysis, computer vision, etc. In social network analysis, an important research topic is to identify cohesive subgroups of individuals within a network where cohesive subgroups are defined as "subsets of actors among whom there are relatively strong, direct, intense, frequent, or positive ties" ([22]). Many approaches, such as clique-based, degree-based, and matrix-perturbation-based, have been proposed to extract cohesive subgroups from social network [22]. Communities also play an important role in Web analysis. For example, Flake et al. [6] defined Web communities as "a set of sites that have more links to members of the community than to non-members", and proposed algorithms to identify Web communities based on a maximum flow/minmum cut framework. Newman et al. [13] defined a metric called modularity measure to quantify the strength of community structure which we will discuss in detail in a later section. In computer vision, community extraction is closely related to image segmentation problem. One effective method in this area is the spectral clustering algorithm [4, 5, 18, 25] where the eigenvectors of certain normalized similarity matrices are used for the clustering purpose. Later, White et al. [23] pointed out the close relationship between Newman's modularity and the spectral clustering and proposed several algorithms to combine the two approaches. Yu et al. [24] proposed a novel clustering framework on graphs where the cluster memberships are assigned in a probabilistic way. In Yu's framework, cluster memberships can be extracted in different resolutions, representing local or global cluster structures. A common issue in all the above studies is that they only analyzed *static networks* where no temporal analysis is used for evolution study. Another issue in these studies is that they treat community extraction as a graph partition problem and therefore always result in hard community memberships, which disallows an individual to participate multiple communities at the same time.

Recently, there exists a growing body of literature on analyzing communities and their evolutions in *dynamic networks*. Kumar et al. [8] studied the evolution of the blogosphere as a graph in terms of the change of characteristics, (such as in-degree, out-degree, strongly connected components), the change of communities, as well as the burstiness in blog community. Leskovec et al. [10] studied the patterns of growth for graphs in various fields and proposed generators that produce graphs exhibiting the discovered patterns. Palla et al [16] analyzed a co-authorship network and a mobile phone network, where both networks are dynamic, by using the clique percolation method (CPM). Toyoda et al. [21] studied the evolution of Web communities from a series of Web achieves by defining different types of community changes, such as emerge, dissolve, grow, and shrink, as well as a set of metrics to quantify such changes for community evolution analysis. Spiliopoulou et al. [19] proposed a framework, MONIC, to model and monitor cluster transitions over time. They defined a set of *external transitions* such as survive, split, disappear, to model transactions among different clusters and a set of *internal transitions*, such as size and location transitions to model changes within a community. Asur et al. [1] introduced a family of events on both communities and individuals to characterize evolution of communities. They also defined a set of metrics to measure the stability, sociability, influence and popularity for communities and individuals. Sun et al. [20] proposed a parameter-free algorithm, GraphScope, to mine time-evolving graphs where the Minimum Description Length (MDL) principle is employed to extract communities and to detect community changes. Mei et al. [12] extracted latent themes from text and used the evolution graph of themes for temporal text mining. All these studies, however, have a common weak point—community extraction and community evolution are analyzed in two separated stages. That is, when communities are extracted at a given timestep, historic community structure, which contains valuable information related to current community structure, is not taken into account.

There are some recent studies on evolutionary embedding and clustering that are closely related to our work. In [17], Sarkar et al. proposed a dynamic method that embeds nodes into latent spaces where the locations of the nodes at consecutive timesteps are regularized so that dramatic change is unlikely. In [2], Chakrabarti et al. proposed the first evolutionary clustering methods where the cluster membership at time t is influenced by the clusters at time *t-1*. Chi et al. [3] extended similar ideas and proposed the first evolutionary spectral clustering algorithms. They used graph cut as a metric for measuring community structures and community evolutions. All these studies differ from our work in that they regularize the current community membership at time t by using historic community membership *indirectly*. In Chakrabarti et al.'s evolutionary hierarchical clustering algorithm, historic community structure affects the tree-node merging step in the current time. In their evolutionary k-means clustering algorithm, historic centroids affect the k-mean process at the current time. In Chi et al.'s algorithms, certain eigenvectors, instead of the community structure, are regularized over time. In the work of Sarkar et al., although the relationship among nodes in latent spaces is preserved over time, the issue of communities are not directly addressed. In contrast, in our proposed framework, the community membership itself is directly regularized over time.

2. FORMULATION

2.1 Notation

First, a note on notations. In this paper, we use lower-case letters, e.g., x, to represent scalars, vector-formed letters, e.g., $\vec{v}$, to represent vectors, and upper-case letters, e.g., W, to represent matrices. Both w_{ij} and $(W)_{ij}$ represent the element at the i-th row and j-th column of W. We use $\mathrm{vec}\,(W)$ to denote the vectorization of W, i.e., stacking the columns of W into a column vector. A subscript t on a variable, e.g., W_t or $w_{t;ij}$, denotes the value of that variable at time t. However, to avoid notation clutter, we try not to use the subscript t unless it is needed for clarity.

We assume that edges in the networked data are associated with discrete timesteps. We use a *snapshot* graph $\mathcal{G}_t(\mathcal{V}_t, \mathcal{E}_t)$ to model interactions at time t where in $\mathcal{G}_t$, each node $v_i \in \mathcal{V}_t$ represents an individual and each edge $e_{ij} \in \mathcal{E}_t$ denotes the presence of interactions between v_i and v_j. Assuming $\mathcal{G}_t$ has n nodes, we use a matrix $W \in \mathcal{R}_+^{n\times n}$ (which is short for W_t) to represent the similarity between nodes in $\mathcal{G}_t$, where $w_{ij} > 0$ if $e_{ij} \in \mathcal{E}_t$ and otherwise $w_{ij} = 0$. Without loss of generality, we assume that $\sum_{i,j} w_{ij} = 1$. Over time, the interaction history is captured by a sequence of snapshot graphs $\langle \mathcal{G}_1, \cdots, \mathcal{G}_t, \cdots \rangle$ indexed by time.

2.2 Basic Formulation

As mentioned in the introduction, we want to analyze communities and their evolutions in a unified process. That is, at time t, we prefer a community structure so that the community evolution from *t-1* to t is not unreasonably dramatic. To achieve this goal, we propose to use the community structure at time *t-1* (already extracted) to regularize the community structure at current time t (to be extracted). To incorporate such a regulation, we introduce a cost function to measure the quality of community structure at time t, where the cost consists of two parts—a *snapshot cost* and a *temporal cost*:

$$cost = \alpha \cdot \mathcal{CS} + (1-\alpha) \cdot \mathcal{CT} \qquad (1)$$

This cost function is first proposed by Chakrabarti et al. [2] in the context of evolutionary clustering. In this cost function, the snapshot cost $\mathcal{CS}$ measures how well a community structure fits W, the observed interactions at time t. The temporal cost $\mathcal{CT}$ measures how consistent the community structure is with respect to historic community structure (at time *t-1*). The parameter α is set by the user to control the level of emphasis on each part of the total cost.

2.2.1 Snapshot Cost

A community structure at time t should fit W well, where W is the observed interaction (similarity) matrix at time t. This requirement is reflected in the snapshot cost $\mathcal{CS}$ in the cost function Eq. (1). We first describe how we model the community structure and how we define the snapshot cost.

Assume there exist m communities at time t. We further assume that the interaction (similarity) w_{ij} is a combined effect due to all the m communities. That is, we approximate w_{ij} using a mixture model $w_{ij} \approx \sum_{k=1}^m p_k \cdot p_{k\to i} \cdot p_{k\to j}$, where p_k is the prior probability that the interaction w_{ij} is due to the k-th community, $p_{k\to i}$ and $p_{k\to j}$ are the probabilities that an interaction in community k involves node v_i and v_j, respectively. Written in a matrix form, we have $W \approx X\Lambda X^T$ where $X \in \mathcal{R}_+^{n\times m}$ is a non-negative matrix with $x_{ik} = p_{k\to i}$ and $\sum_i x_{ik} = 1$. In addition, Λ is an $m \times m$ non-negative diagonal matrix with $\lambda_k = p_k$, where λ_k is short for λ_{kk}. Matrices X and Λ (or equivalently, their product $X\Lambda$) fully characterize the community structure in the mixture model. This model was first proposed by Yu et al. in [24].

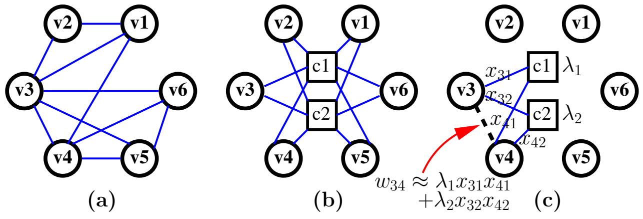

Figure 1: Schematic illustration of soft communities: (a) the original graph, (b) the bipartite graph with two communities, and (c) how to approximate an edge (w_{34})

In Fig. 1, we use a toy example with 6 nodes and 2 communities to illustrate this model of community structure. For a general graph (a), we use a special bipartite graph (b) to approximate (a). Note that (b) has two more nodes, i.e., c_1 and c_2, corresponding to the two communities. However, because (b) is a bipartite graph (i.e., an edge can only occur between a node v and a community c), it has less degree of freedom and so it is a more parsimonious explanation of (a). In (c), we show how an edge w_{34} is generated in the mixture model as the sum of $\lambda_1 x_{31} x_{41}$ and $\lambda_2 x_{32} x_{42}$. Equivalently, we are approximating W, which has rank n, by a product in the form of $X\Lambda X^T$, which has rank m. Based on this model, we define the snapshot cost $\mathcal{CS}$ as the error introduced by such an approximation, i.e.,

$$\mathcal{CS} = D(W \| X\Lambda X^T)$$

where $D(A\|B) = \sum_{i,j}(a_{ij}\log\frac{a_{ij}}{b_{ij}} - a_{ij} + b_{ij})$ is the KL-divergence between A and B. So the snapshot cost is high when the approximate community structure $X\Lambda X^T$ fails to fit the observed data W well.

2.2.2 Temporal Cost

In the cost function Eq. (1), the temporal cost $\mathcal{CT}$ is used to regularize the community structure so that it is less probable for unreasonably dramatic community evolution from time *t-1* to t. We propose to achieve this regularization by defining $\mathcal{CT}$ as the difference between the community structure at time t and that at time *t-1*. Recall that the community structure is captured by $X\Lambda$. Therefore, with $Y \doteq X_{t-1}\Lambda_{t-1}$, the temporal cost is defined as

$$\mathcal{CT} = D(Y \| X\Lambda)$$

where D is the KL-divergence as defined before. So the temporal cost $\mathcal{CT}$ is high when there is a dramatic change of community structure from time *t-1* to t.

2.2.3 Putting It Together

Putting the snapshot cost $\mathcal{CS}$ and the temporal cost $\mathcal{CT}$ together, we have an optimization problem as to find the best community structure at time t, expressed by X and Λ, that minimizes the following total cost

$$cost = \alpha \cdot D(W\|X\Lambda X^T) + (1-\alpha)\cdot D(Y\|X\Lambda) \quad (2)$$

subject to $X \in \mathcal{R}_+^{n\times m}$, $\sum_i x_{ik} = 1$, and Λ being an m-by-m non-negative diagonal matrix. Solving this optimization problem is the core of our *FacetNet* framework.

2.3 Justification

We now provide two interpretations to the cost function Eq. (2), one from the point of view of information theory and the other from that of probabilistic generative model.

2.3.1 An Information Theory Interpretation

In information theory, the KL-divergence $D(P\|Q)$ is also known as the relative entropy, and it represents the information gain if we use the precise distribution P instead of the approximate model Q (where Q tries to model P). In our community structure, $X\Lambda X^T$ is the marginal distribution induced from the bipartite model and it tries to approximate W. As a result, $D(W\|X\Lambda X^T)$ gives us the information gain (or the error introduced) from our community structure $X\Lambda X^T$ to the true distribution W. A higher information gain suggests a larger error introduced by $X\Lambda X^T$ and therefore implies a higher snapshot cost $\mathcal{CS}$.

Similarly, in $D(Y\|X\Lambda)$, Y represents the community structure at time *t-1*. When we try to use the current community structure $X\Lambda$ to explain Y, if the information gain from $X\Lambda$ to Y is larger, then the change of community structure from time *t-1* to t will be more dramatic, and therefore the temporal cost $\mathcal{CT}$ will be higher.

2.3.2 A Probabilistic Interpretation

Next we provide a probabilistic interpretation of our framework by using a first-order Markov generative model. The basic ideas are that (1) the currently observed data W_t is generated from the current community structure following a certain distribution and (2) the current community structure at time t is generated by using the community structure at time *t-1* as the prior distribution.

Let U_t be the community parameters at time t and U_{t-1} be those at time *t-1*. The goal is then to estimate the unseen parameters U_t, given W_t and U_{t-1}, i.e.,

$$U^* = \arg\max_{U_t} \log P(W_t, U_t | U_{t-1})$$

We further assume a first-order Markov model, as illustrated in Fig. 2, and we have $P(W_t, U_t|U_{t-1}) = P(W_t|U_t)P(U_t|U_{t-1})$. Therefore the log-likelihood function can be written as

$$L(U_t) = \log P(W_t|U_t) + \log P(U_t|U_{t-1}) \quad (3)$$

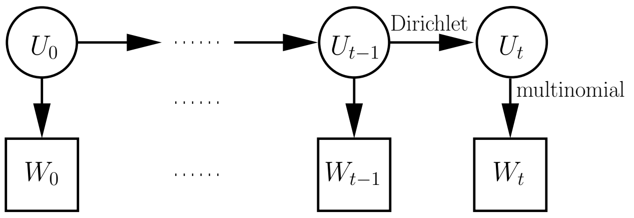

Figure 2: Schematic illustration of the probabilistic model, where $U = (X, \Lambda)$

In our model, $U_{t-1} = (X_{t-1}, \Lambda_{t-1})$ and $U_t = (X_t, \Lambda_t)$. Under the above probabilistic model, we have the following theorem, whose proof is given in the Appendix.

THEOREM 1. *Under the assumptions that*

(1) vec (W_t) *follows a multinomial distribution with parameter* θ_t*, where* $\theta_t = \text{vec}\left(X_t\Lambda_t X_t^T\right)$*, and*

(2) vec $(X_t\Lambda_t)$ *follows a Dirichlet distribution with parameter* ϕ_t*, where* $\phi_t = \nu\text{vec}(X_{t-1}\Lambda_{t-1}) + 1$ *and* $\nu = (1-\alpha)/\alpha$*;*

the parameter estimation of X_t *and* Λ_t *by maximum a posterior (MAP) in Eq.* (3) *is equivalent to that by minimizing the cost function in Eq.* (2)*.*

2.4 Solution

In this subsection, we first provide an iterative algorithm to solve the optimization problem defined by Eq. (2) and then show the time complexity of our algorithm.

2.4.1 An Iterative Algorithm

In our algorithm, we use the following update rules and as stated in the following theorem, in each iteration, the algorithm updates the values of X and Λ in such a way that the cost function defined in Eq. (2) is monotonically decreased.

THEOREM 2. *The following update rules will monotonically decrease the cost function defined in Eq.* (2) *and therefore converge to an optimal solution to the objective function:*

$$x_{ik} \leftarrow x_{ik} \cdot 2\alpha \cdot \sum_j \frac{w_{ij} \cdot \lambda_k \cdot x_{jk}}{(X\Lambda X^T)_{ij}} + (1-\alpha) \cdot y_{ik} \qquad (4)$$

then normalize such that $\sum_i x_{ik} = 1, \forall k$

$$\lambda_k \leftarrow \lambda_k \cdot \alpha \cdot \sum_{ij} \frac{w_{ij} \cdot x_{ik} \cdot x_{jk}}{(X\Lambda X^T)_{ij}} + (1-\alpha) \cdot \sum_i y_{ik} \qquad (5)$$

then normalize such that $\sum_k \lambda_k = 1$.

The proof for the correctness and the convergence of the above update rules is skipped due to space limit.

2.4.2 *Time Complexity*

We now show the time complexity for each iteration of the updates in Theorem 2. The most time-consuming part is to compute $(X\Lambda X^T)_{ij}$ for all $i, j \in \{1, \dots, n\}$. However, it turns out that we do not have to compute $(X\Lambda X^T)_{ij}$ for each pair of (i, j), thanks to the sparseness of W. In W, the number of non-zero elements is the number of edges in the snapshot graph, which we denote by ℓ. Then for each non-zero w_{ij}, we compute the corresponding $(X\Lambda X^T)_{ij}$, which takes $O(m)$ time with m being the number of communities. As a result, the total complexity is $O(\ell m)$. If we consider m, the number of communities, to be a constant and if the degree of nodes in the snapshot graph is bounded by another constant, then the complexity is reduced to $O(n)$, i.e., linear in the number of nodes in the snapshot graph.

2.5 Communities and Their Evolutions

After obtaining the solution to Eq. (2) by using our algorithm in Theorem 2, here are the ways we utilize the solution to analyze communities and their evolutions.

2.5.1 *Community Membership*

Assume we have computed the result at time *t-1*, i.e., (X_{t-1}, Λ_{t-1}), and the result at time t, i.e., (X_t, Λ_t). In addition, we define a diagonal matrix D_t, whose diagonal elements are the row sums of $X_t\Lambda_t$, i.e., $d_{t;ii} = \sum_j (X_t\Lambda_t)_{ij}$. Then we claim that the i-th row of $D_t^{-1}X_t\Lambda_t$ indicates the soft community memberships of v_i at time t. We illustrate this by using an example shown in Fig. 3. Recall that in the bipartite graph at time t (the right side of Fig. 3(a)), the weights of edges connecting v_i to c_1, c_2, and c_3 represent the joint probability $P(v_i, c_1)$, $P(v_i, c_2)$, and $P(v_i, c_3)$. The D_t^{-1} part normalizes this joint probability to get $P(c_1|v_i)$, $P(c_2|v_i)$, and $P(c_3|v_i)$, i.e., the conditional probability that v_i belongs to c_1, c_2, and c_3, respectively. And this conditional probability is exactly the soft community membership we are looking for. Furthermore, we can see that the i-th diagonal element of D_t provides information about the level of activity of v_i at time t.

2.5.2 *Community Net*

The community structure itself, on the other hand, is expressed by $\Lambda_t X_t^T D_t^{-1} X_t \Lambda_t$. For this we again look at the bipartite graph at time t (the right side of Fig. 3(a)). Induced from this bipartite graph, $X_t\Lambda X_t^T$ gives a marginal distribution on the subgraph with nodes $\{v_1, \dots, v_6\}$ in order to approximate W_t. In a dual fashion, also induced from

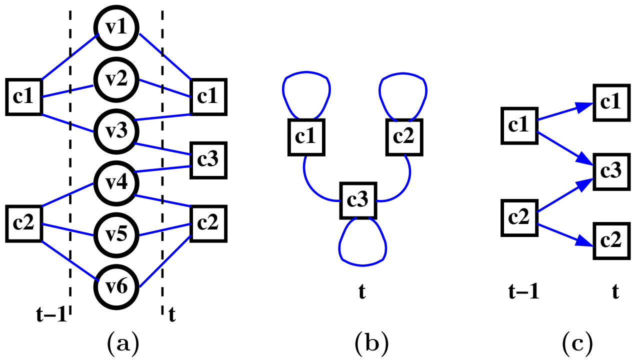

Figure 3: Schematic illustration of communities and their evolutions: (a) two bipartite graphs at time *t-1* and time t (merged by v_i's), (b) the Community Net at time t induced by the bipartite graph at time t, and (c) the Evolution Net from *t-1* to t induced by the two bipartite graphs

this bipartite graph, $\Lambda_t X_t^T D_t^{-1} X_t \Lambda_t$ gives a marginal distribution on the subgraph with nodes $\{c_1, c_2, c_3\}$ (Fig. 3(b)) and this is exactly the community structure we are looking for. We call this induced subgraph on the community nodes (i.e.,$\{c_1, c_2, c_3\}$) a *Community Net*. Note that to induce the community net, each node v_i participates in all the communities, with different levels. This is more reasonable than traditional methods in which each node can only contribute to a single community.

2.5.3 *Evolution Net*

To derive the community evolutions, we align the two bipartite graphs, that at time *t-1* and that at time t, side by side by merging the corresponding network nodes v_i's, as illustrated in Fig. 3(a). Then a natural definition of community evolution (from $c_{t-1;i}$ at time *t-1* to $c_{t;j}$ at time t) is the probability of starting from $c_{t-1;i}$, walking through the merged bipartite graphs, and reaching $c_{t;j}$. Such a walking process produces what we call the *Evolution Net* to represent community evolutions, as illustrated in Fig. 3(c). A simple derivation shows that $P(c_{t-1;i}, c_{t;j}) = (\Lambda_{t-1} X_{t-1}^T D_t^{-1} X_t \Lambda_t)_{ij}$ and $P(c_{t;j}|c_{t-1;i}) = (X_{t-1}^T D_t^{-1} X_t \Lambda_t)_{ij}$. Again, each node and each edge contribute to the evolution from $c_{t-1;i}$ to $c_{t;j}$. That is, all individuals and all interactions are related to all the community evolutions, with different levels. We believe this is more reasonable than how community evolutions are derived in traditional methods. In tradition methods, usually the intersection and union of *community members* at different time are used alone to compute community evolutions, with a questionable assumption that all members in a community should be treated with identical importance.

3. EXTENSIONS

In this section we introduce two extensions to our basic framework in order to handle inserting and removing of individuals and to determine the number of communities in a dynamic network over time.

3.1 Inserting and Removing Nodes

In real applications, it occurs very often that some new individuals join a dynamic network (e.g., a new author in a paper co-authorship network) and existing ones leave (e.g., a blogger who stops blogging). We provide the following heuristic techniques in our algorithm to handle such inserting and removing of nodes.

Assume that at time t, out of the n nodes in the network, n_1 existing nodes are removed from and n_2 new nodes are inserted into the network. We first handle the n_1 removed nodes by removing the corresponding n_1 rows from Y in Equations (4) and (5) to get Y'. Next, we scale Y' to get Y'' so that Y'' is a valid joint distribution, i.e., $Y'' = Y'/\sum_{ij} y'_{ij}$. The basic idea behind this heuristic is that we assume the n_1 nodes are randomly selected, independent of their community membership. Under such an assumption, Y'' is a conditional distribution, conditioning on the remaining $n - n_1$ nodes in the network. To add the n_2 nodes, we pad n_2 rows of zeros to Y'' to get $\hat{Y}$. This heuristic is actually equivalent to assuming that these n_2 nodes have already existed at time *t-1* but as isolated nodes.

3.2 Changing Community Numbers

So far we have assumed that the number of communities, m, is given beforehand by the user. However, such an assumption will limit the scope of application of our framework. In this subsection we try to answer two questions: how to automatically determine the number of communities at a given time t and how to revise our framework to allow the number of communities to change in different timesteps.

3.2.1 Soft Modularity

In [13], Newman et al. introduces an elegant concept, the *modularity* Q, to measure the goodness of a community partition $\mathcal{P}_m$ where Q is defined as

$$Q(\mathcal{P}_m) = \sum_{k=1}^{m} \left[\frac{\mathcal{A}(V_k, V_k)}{\mathcal{A}(V, V)} - \left(\frac{\mathcal{A}(V_k, V)}{\mathcal{A}(V, V)} \right)^2 \right] \tag{6}$$

with $\mathcal{A}(V_p, V_q) = \sum_{i \in V_p, j \in V_q} w_{ij}$. Basically, Q measures the deviation between the chance for edges among communities to be generated due to the community structure and the chance for the edges to be generated randomly. Extensive experimental results [13, 23] have demonstrated that Q is an effective measure for the community quality, where a maximal Q is a good indicator of the best community structure and therefore the best community number m.

Here we extend the concept of modularity to handle soft membership by defining a *Soft Modularity* Q_s:

$$\begin{aligned} Q_s = & Tr\left[(D^{-1}X\Lambda)^T W (D^{-1}X\Lambda) \right] \\ & - \vec{1}^T W^T (D^{-1}X\Lambda)(D^{-1}X\Lambda)^T W \vec{1} \end{aligned} \tag{7}$$

where $\vec{1}$ is a vector whose elements are all ones. Q_s has the following nice property, whose proof is given in the Appendix.

Theorem 3. *The Q_s defined in Eq. (7) has the same probabilistic interpretation as the Q defined in Eq. (6), but in the context of soft community membership. In addition, Q_s is a generalized modularity in that Q_s is identical to Q when $D^{-1}X\Lambda$ becomes a hard community membership (i.e., each row of $D^{-1}X\Lambda$ has one 1 and m-1 zeros).*

So to detect the best community number m at time t, we run our algorithm for a range of candidates for m and pick the best one determined by Q_s.

3.2.2 Extended Formulas

If we allow different community numbers at time t and *t-1*, then we have to revise Eq. (2) accordingly because in Eq. (2), the term $D(Y\|X\Lambda)$ requires Y (the community structure at *t-1*) and $X\Lambda$ (the community structure at t) to have the same number of columns and therefore the same number of communities. To solve this issue, we first define $Z \doteq X_{t-1}\Lambda_{t-1}X_{t-1}^T$ and then revise the cost function to be

$$cost = \alpha \cdot D(W\|X\Lambda X^T) + (1-\alpha) \cdot D(Z\|X\Lambda X^T) \tag{8}$$

The basic idea is that when the community numbers are different at time t and *t-1*, instead of regularizing the community structure itself, we regularize the marginal distribution induced by the community structure at time t (i.e., $X\Lambda X^T$, which approximates W_t) so that it is not too far away from that at time *t-1* ($X_{t-1}\Lambda_{t-1}X_{t-1}^T$). For the cost function given in Eq. (8), the following update rules are used.

Theorem 4. *The following update rules will monotonically decrease the cost function defined in Eq. (8) and therefore converge to an optimal solution to the objective function.*

$$x_{ik} \leftarrow x_{ik} \cdot \sum_j \frac{(\alpha \cdot w_{ij} + (1-\alpha) \cdot z_{ij}) \cdot \lambda_k \cdot x_{jk}}{(X\Lambda X^T)_{ij}} \tag{9}$$

$$\textit{then normalize such that } \sum_i x_{ik} = 1, \forall k$$

$$\lambda_k \leftarrow \lambda_k \cdot \sum_{ij} \frac{(\alpha \cdot w_{ij} + (1-\alpha) \cdot z_{ij}) \cdot x_{ik} \cdot x_{jk}}{(X\Lambda X^T)_{ij}} \tag{10}$$

$$\textit{then normalize such that } \sum_k \lambda_k = 1.$$

The proof for the correctness and the convergence of the above update rules is skipped due to space limit.

4. EXPERIMENTAL STUDIES

In this section, we use several synthetic datasets, a blog dataset, and a paper co-authorship dataset to study the performance of our *FacetNet* framework.

4.1 Synthetic Datasets

4.1.1 Synthetic Dataset # 1

We start with the first synthetic dataset, which is a *static* network, to illustrate some good properties of our framework. This dataset was first studied by White et al. [23] and is shown in Fig. 4(a). The network contains 15 nodes which roughly form 3 communities—$C1$, $C2$, and $C3$— where edges tend to occur between nodes in the same community. We first check our soft modularity measure. We apply our algorithm to the network with various community numbers m and the resulting Q_s values are plotted in Fig. 4(b). In addition, in Fig. 4(b) we also show the modularity values Q that are reported by White et al. in [23]. As can be seen from the plot, both Q_s and Q show distinct peaks when $m = 3$, which corresponds to the correct community number.

Next, after our algorithm correctly partitions the 15 nodes into three communities, we illustrate the soft community membership by studying two communities among the three—$C1 = \{6, 7, 8, 9, 10\}$ and $C2 = \{11, 12, 13, 14, 15\}$. In Fig. 4(a), we use the same circle shape to represent these 10 nodes but use different gray levels to indicate their community

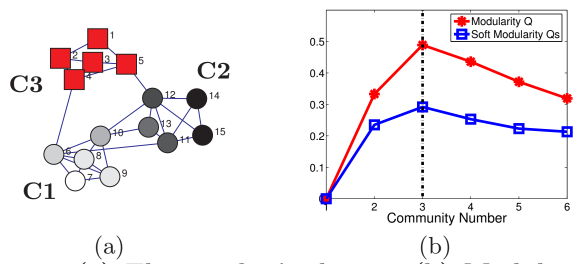

Figure 4: (a) The synthetic dataset (b) Modularity and soft modularity under different community numbers

membership—we use white color to illustrate the level that a node belongs to $C1$ and dark color to show the level that a node belongs to $C2$. As can be seen, while node 7, node 14, and node 15 have very clear community memberships, node 10 and node 13, who are on the boundary between $C1$ and $C2$, have rather fuzzy membership. That is, our algorithm is capable of assigning meaningful soft membership to a node to indicate to which level the node belongs to a certain community.

4.1.2 Synthetic Dataset # 2

The second dataset is generated according to the description by Newman et al. in [13]. This dataset contains 128 nodes, which are divided 4 communities of 32 nodes each. We generate data for 10 consecutive timesteps. In each timestep from 2 to 10, dynamics are introduced in the following way: from each community we randomly select 3 members to leave their original community and to join randomly the other three communities. Edges are added randomly with a higher probability p_{in} for within-community edges and a lower probability p_{out} for between-community edges. However, the average degree for the nodes is set to 16. As a result, a single parameter z, which represents the mean number of edges from a node to nodes in other communities, is enough to describe the data.

Because we have the ground truth for the community membership at each timestep, we directly study the accuracy of the community structure obtained by our framework. We compare our *FacetNet* framework with 3 baseline algorithms. The first baseline, which we call *EvolSpec*, is the evolutionary spectral clustering algorithm proposed by Chi et al. [3]. Because *EvolSpec* is an evolutionary version of the Normalized Cut (*NCut*) algorithm by Shi et al. [18], we take *NCut* as our second baseline. Similarly, *FacetNet* is essentially an evolutionary version of the soft clustering method (*SNMF*) by Yu et al. [24], we take *SNMF* as our third baseline. Notice that *FacetNet* and *EvolSpec* are evolutionary algorithms whereas *NCut* and *SNMF* are not—*NCut* and *SNMF* work on each snapshot graph independently of other snapshot graphs. In addition, to make the results comparable, for *FacetNet* and *SNMF* we convert the soft membership into 0/1 indicators by assigning each node to the community it most likely belongs to. Furthermore, in all the experiments, for *FacetNet* and *EvolSpec* we set α to be 0.9. Fig. 5(a) and 5(b) show the accuracy and standard error of the community membership obtained by the four algorithms for two datasets generated with $z = 3$ and $z = 5$, respectively. The accuracy is computed by the mutual information between the derived community membership and the ground truth, where a higher mutual information indicates better accuracy. From the figures we can see that when $z = 3$, i.e., when there is less noise and hence the community structure is easy to detect, both *FacetNet* and *EvolSpec* have accuracy improved starting from the second timestep, which suggests that an evolution framework is beneficial in this dynamic network. In comparison, *NCut* and *SNMF* have relatively flat accuracy over all timesteps, with *NCut* slightly outperforms *SNMF*. For the data where $z = 5$, there are more edges going between communities and therefore the community structure is more difficult to detect. From Fig. 5(b) we can see that although at the first few timesteps *FacetNet* does not perform as good as *EvolSpec*, as time going further, *FacetNet* starts to outperform *EvolSpec*. This suggests that the benefits of *FacetNet* accumulates over time more than *EvolSpec*. In addition, as for the case with $z = 3$, when $z = 5$, both *FacetNet* and *EvolSpec* clearly outperform their non-evolutionary version, *NCut* and *SNMF*.

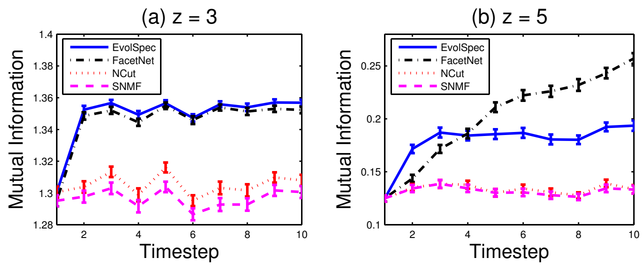

Figure 5: Mutual information with respect to the ground truth over 10 timesteps when (a) $z = 3$ and (b) $z = 5$

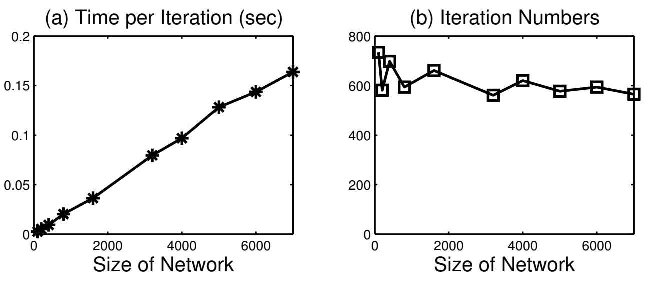

Figure 6: Running time for networks of different sizes (a) time per iteration (sec) and (b) number of iterations until converge

Next, we study the time performance of *FacetNet*. We repeat the above experiment over a family of networks of various sizes (node numbers). In Fig. 6(a) we show the average running time per iteration of our algorithm on networks with different sizes. In 6(b) we showed the number of iteration needed for convergence when the convergence criterion is that the change between two consecutive iterations is below a threshold of 1e-5. As can be seen, first, the running time per iteration scales linearly with the size of network, which validates our theoretical analysis in Section 2; second, the number of iteration needed for convergence is insensitive to the network size, which implies that the overall running time of our algorithm scales linearly to the size of network.

4.2 NEC Blog Dataset

The blog data was collected by an NEC in-house blog crawler. Given seeds of manually picked highly ranked blogs, the crawler discovered blogs that are densely connected with the seeds, resulting in an expanded set of blogs that communicate with each other. The crawler then continued monitoring for new entries over a long time period. This NEC blog data set contains 148,681 entry-to-entry links among 407 blogs crawled during 12 consecutive month (a timestep is one month), between August 2005 and September 2006. Following [14], we define W by $w_{ij} = \tilde{w}_{ij} / \sum_{p,q} \tilde{w}_{pq}$ where $\tilde{w}_{ii} = 1$, $\tilde{w}_{ij} = \exp(-1/(\gamma \cdot l_{ij}))$ if $e_{ij} \in \mathcal{E}_t$, and otherwise $\tilde{w}_{ij} = 0$. In the formula, l_{ij} is the edge weight (e.g., # of links) of e_{ij} and γ, which is set to 0.2, is a parameter to control marginal effect when l_{ij} is increased.

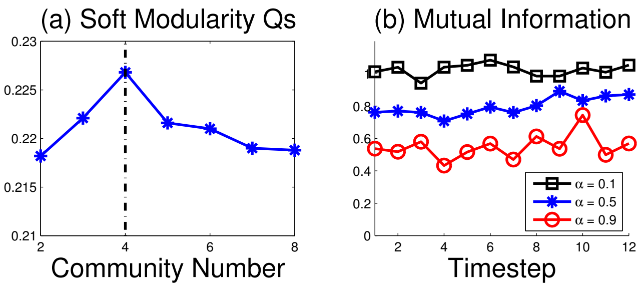

Figure 7: (a) Soft modularity and (b) mutual information under different α for the NEC dataset

We start with analyzing the overall picture of the dataset. We first aggregate all the edges over all timesteps into a single network and apply our algorithm to compute the soft modularity score Q_s under different community numbers. As can be seen in Fig. 7(a), a clear peak shows when the community number is 4. We draw the aggregated graph in Fig. 8, according to the main community each blog most likely belongs to. In addition, in Table 1 we list the top keywords, measured by the *tf-idf* score, that occur in the posts of these four communities. It seems that $C1$ focuses on technology, $C2$ on politics, $C3$ on international entertainment, and $C4$ on digital libraries.

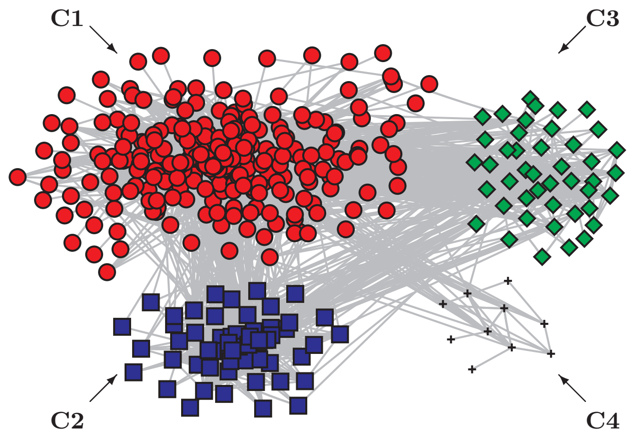

Figure 8: Four communities in the NEC dataset

Next, we analyze the blog data as a dynamic network. After studying the content of the blogs, we find that the above

Table 1: Top keywords among the four communities in the NEC dataset, sorted by the *tf-idf* score

C1	adsense, beta, skype, firefox, msn, rss, aol, yahoo, google, ebay, desktop, wordpress, voip, feeds, myspace, podcasting, technorati, search, engine, browser, ads, gmail, windows, os, developer, venture, marketing, apple, podcasts, developers, engines, mac, publishers, ceo, linux
C2	gop, uranium, hezbollah, democrats, rove, cia, republicans, saddam, qaeda, tax, republican, iraqi, roberts, bush, clinton, iraq, senate, troops, terrorists, administration, terrorist, wilson, conservative, taxes, liberal, intelligence, israel, terror, iran, weapons, war, soldiers
C3	shanghai, robots, installation, japan, japanese, architecture, art, chinese, china, saudi, phones, filed, mobile, games, korea, rfid, sex, green, camera, sound, cell, body, africa, phone, entertainment, film, gay, india, fuel, archive, design, elections, flash, device, water, wireless, south
C4	library, learning, digital, resources, collection, conference, staff, communities, students, session, books, database, access, survey, university, science, canada, myspace, articles, education, technologies, knowledge, filed, virtual, tools, research, david, learn, services, flickr, computers

four communities stay rather stable over all the timesteps. This effect is partially due to the way these blogs are selected by our focused crawler—our crawler chose to crawl a densely connected subgraph of the blogosphere where each node in the subgraph has large number of links and high level of interaction intensities. Therefore, most of the selected blogs belong to some well-known bloggers and they seldom move around between communities. We apply our *FacetNet* algorithm on the data with different α. And we compute the mutual information between the extracted communities at each timestep and the four communities shown in Fig. 8. Fig. 7(b) shows the results under α=0.1, 0.5, and 0.9. As can be seen, as α increases, our algorithm emphasizes less on the temporal smoothness and as a result, the community structure has higher variation over time. In addition, as α increases, the communities at each timestep deviate further from the communities obtained from the aggregated data. These results on one hand justify our arguments in the introduction section and on the other hand demonstrate that our *FacetNet* framework is capable of controlling the tradeoff between the snapshot cost and the temporal cost in the cost function Eq. (1).

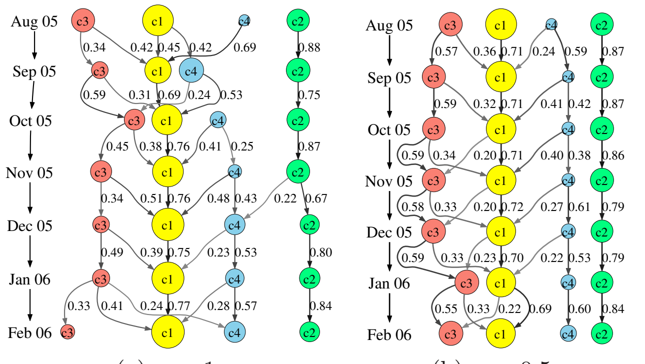

Figure 9: The Evolution Net of the NEC dataset when (a) $\alpha = 1$ and (b) $\alpha = 0.5$

Fig. 9 shows the Evolution Net derived from our framework when α=1 and 0.5 ($\alpha = 1$ means no temporal smoothness is considered). In the Evolution Net, the size of a node is proportional to λ_k and it represents the size of the corresponding community. The edge label indicates the probability of a transition from the source community at *t-1* to the

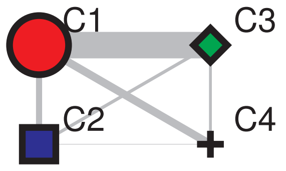

Figure 10: The Community Net for the NEC dataset in September 2005

target community at t. (To avoid clutter, we did not show edges with probabilities less than 0.2). From Fig. 9(a) we see that when no temporal smoothness is considered, $C4$ disappeared at the second timestep (Sep 05) and re-appeared in the third timestep (Oct 05). However, by carefully examining the original data, we did not find any significant events so support such changes. Therefore, we conjecture that these changes are due to the data noises at the second timestep, which triggered the algorithm to split $C1$ into two communities and merge $C4$ to one of them. In comparison, as can be seen from Fig. 9(b), when there is a temporal smoothness term, the four communities remain relatively stable. That is, although there exist transitions among different types of communities, the majority of transitions are between communities of the same type. These results demonstrate that the *FacetNet* framework is more robust to data noise.

From the Evolution Net, we can also obtain some other observations. For example, the political community $C2$ is rather isolated from the rest communities over all the time. In comparison, both $C3$ and $C4$ interacts with $C1$ heavily. In addition, in Fig. 10 we show the Community Net at an arbitrary timestep (Sep 05). In the Community Net, the node sizes are proportional to $\lambda_{t;k}$ and the edge weights are proportional to the corresponding entries in $\Lambda_t X_t^T D_t^{-1} X_t \Lambda_t$ (self-loops are not shown). We can see that this Community Net is a good synopsis of the aggregated network in Fig. 8.

4.3 DBLP Co-authorship Dataset

The DBLP co-authorship dataset is a subset of that used by Asur et al. [1]. It contains papers in 28 conferences over 10 years (1997–2006), which span three areas—database, data mining, and artificial intelligence. We selected a dense subgraph of 2950 authors from the original dataset and partition the time into three periods with overlap: 1997–2000, 2000–2003, 2003–2006.

Due to the space limit, we skip the detailed performance study and only point out two interesting issues about this dataset. First, in the first two periods, the soft modularity shows (local) peaks at $m = 4$ while in the third period, $m = 3$ is optimal. As a result, for this experiment we used the update algorithm described in Theorem 4. Second, in Table 2 we list the top authors in the three communities detected by *FacetNet* in the third period (2003–2006). Here the rank is determined by the value x_{ik}, i.e., $p_{k\to i}$. Recall that $p_{k\to i}$ indicates to what level the k-th community involves the i-th node. So from our framework, we can directly infer who are the important members in each community. However, notice that by *important*, we are not judging the quality or quantity of papers by an author. Instead, in our framework the importance of a node in a community is determined by its contribution to the community structure. For example, in Table 2, some of the top authors in the AI area are ranked high partially because they participated in a series of papers with up to 20 co-authors and these large co-author cliques heavily influenced the community structure of the AI community in our dataset.

Table 2: Top members in the three communities during 2003–2006, sorted by x_{ik}, i.e., $p_{k\to i}$

Data Mining	Database	Artificial Intelligence
Philip S. Yu	Divesh Srivastava	Hans-Peter Kriegel
Jiawei Han	Surajit Chaudhuri	William C. Regli
Wei Wang	Nick Koudas	Maxim Peysakhov
Jian Pei	Elke A. Rundensteiner	Vincent A. Cicirello
Divyakant Agrawal	Jennifer Widom	Evan Sultanik
Kian-Lee Tan	Raghu Ramakrishnan	Gustave Anderson
Beng Chin Ooi	Jeffrey F. Naughton	Andrew Burnheimer
Stanley B. Zdonik	David J. DeWitt	David Dorsey
Nikos Mamoulis	Rajeev Motwani	Moshe Kam
Walid G. Aref	H. V. Jagadish	Joseph Kopena

5. CONCLUSION

The analysis of communities and their evolutions in dynamic temporal networks is a challenging research problem with broad applications. In this paper, we proposed a framework, *FacetNet*, to solve this problem. Unlike traditional two-stage techniques that separate the task of community extraction and the task of evolution extraction, the *FacetNet* framework combines these two tasks in a unified process. Therefore, not only community structures determine the evolutions, but also the historic evolution patterns regularize current community structure. As a result, it is less likely for the current community structure to deviate too dramatically from the most recent history. In addition to the basic framework, we also proposed to use soft community membership and we introduced several novel concepts such as *Community Net*, *Evolution Net*, and *Soft Modularity*, to measure and to visualize the resulting communities and their evolutions. Extensive experimental studies demonstrated that our framework provide communities and evolutions that are more accurate and more robust to data noise.

Acknowledgments

We thank Kai Yu for providing us with the SNMF source code and for the helpful discussion; thank Junichi Tatemura and Koji Hino for helping prepare the blog dataset; thank Sitaram Asur and Srinivasan Parthasarathy for providing us with the DBLP dataset.

6. REFERENCES

[1] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. In *Proc. of the 13th ACM SIGKDD Conference*, 2007.

[2] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *Proc. of the 12th ACM SIGKDD Conference*, 2006.

[3] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. In *Proc. of the 13th ACM SIGKDD Conference*, 2007.

[4] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[5] I. S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proc. of the 10th ACM SIGKDD Conference*, 2004.

[6] G. Flake, S. Lawrence, and C. Giles. Efficient identification of web communities. In *Proc. of the 6th ACM SIGKDD Conference*, 2000.

[7] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. of the ACM*, 46(5), 1999.

[8] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins. On the bursty evolution of blogspace. In *Proc. of the 12th WWW Conference*, 2003.

[9] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *Proc. of the 12th ACM SIGKDD Conference*, 2006.

[10] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *Proc. of the 11th ACM SIGKDD Conference*, 2005.

[11] Y. Lin, H. Sundaram, Y. Chi, J. Tatemura, and B. L. Tseng. Blog community discovery and evolution based on mutual awareness expansion. In *Proc. of the Int. Conf. on Web Intelligence*, 2007.

[12] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. In *Proc. of the 11th ACM SIGKDD Conference*, 2005.

[13] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 2004.

[14] H. Ning, W. Xu, Y. Chi, Y. Gong, and T. Huang. Incremental spectral clustering with application to monitoring of evolving blog communities. In *SIAM Int. Conf. on Data Mining*, 2007.

[15] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. In *Technical report, Stanford Digital Library Technologies Project, Stanford University, Stanford, CA, USA*, 1998.

[16] G. Palla, A.-L. Barabasi, and T. Vicsek. Quantifying social group evolution. *Nature*, 446, 2007.

[17] P. Sarkar and A. W. Moore. Dynamic social network analysis using latent space models. *SIGKDD Explor. Newsl.*, 7(2), 2005.

[18] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8), 2000.

[19] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult. Monic: modeling and monitoring cluster transitions. In *Proc. of the 12th ACM SIGKDD Conference*, 2006.

[20] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. GraphScope: parameter-free mining of large time-evolving graphs. In *Proc. of the 13th ACM SIGKDD Conference*, 2007.

[21] M. Toyoda and M. Kitsuregawa. Extracting evolution of web communities from a series of web archives. In *HYPERTEXT '03: Proc. of the 14th ACM conference on hypertext and hypermedia*, 2003.

[22] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[23] S. White and P. Smyth. A spectral clustering approach to finding communities in graph. In *SDM*, 2005.

[24] K. Yu, S. Yu, and V. Tresp. Soft clustering on graphs. In *NIPS*, 2005.

[25] H. Zha, X. He, C. H. Q. Ding, M. Gu, and H. D. Simon. Spectral relaxation for k-means clustering. In *NIPS*, 2001.

APPENDIX

Proof for Theorem 1

PROOF. Given that $X_t \in \mathcal{R}_+^{n\times m}$, each column of which sums up to one, and Λ_t is an m-by-m diagonal matrix and sum to one, we have

$$P(U_t|U_{t-1}) = P(Y_t|U_{t-1}),$$

where $Y_t = X_t\Lambda_t$. Because of the constraint, for any given Y_t, there is a unique pair of X_t and Λ_t such that $Y_t = X_t\Lambda_t$, and X_t and Λ_t satisfy the constraint. Thus, Y_t uniquely determines U_t, which implies

$$P(U_t|U_{t-1}) = P(Y_t|U_{t-1}).$$

The logarithm of MAP is

$$\begin{aligned}
L(U_t) &= \log P(W_t|U_t)P(U_t|U_{t-1}) \\
&= \log P(W_t|U_t)P(X_t\Lambda_t|U_{t-1}) \\
&= \log \mathbf{multinom}(\mathrm{vec}\,(W_t)\,;\theta_t) \\
&\quad + \log \mathbf{Dirichlet}(\mathrm{vec}\,(X_t\Lambda_t)\,;\phi_t) \\
&= \log \frac{(\sum_{ij} W_{t;ij})!}{\prod_{ij} W_{t;ij}!} \prod_{ij} \theta_{t;ij}^{W_{t;ij}} \\
&\quad + \log \frac{1}{B(\phi)} \prod_{ik} Y_{t;ik}^{\nu Y_{t-1;ik}} \\
&= \sum_{ij} W_{t;ij} \log \theta_{t;ij} + \sum_{ik} \nu Y_{t-1;ik} \log Y_{t;ik} + \text{const.}
\end{aligned}$$

Since $\sum_{ij} \theta_{t;ij} = \sum_k \Lambda_{t;kk} = 1$ and $\sum_{ik} Y_{t;ik} = \sum_k \Lambda_{t;kk} = 1$, we can further derive

$$\begin{aligned}
L(U_t) &= -\,D(W_t\|\theta_{t;ij}) - \nu D(Y_{t-1}\|Y_t) + \text{const.} \\
&= -\frac{1}{\alpha}[\alpha D(W_t\|X_t\Lambda_t X_t^T) \\
&\quad + (1-\alpha)D(Y_{t-1}\|X_t\Lambda_t)] + \text{const.}
\end{aligned}$$

Thus, maximizing $L(U_t)$ is equivalent to minimizing the cost function in Eq. (2). □

Proof for Theorem 3

PROOF. In the standard Q formula Eq. (6), the first term $\frac{\mathcal{A}(V_k,V_k)}{\mathcal{A}(V,V)}$ is the empirical probability that a randomly selected edge has both ends in community k. For our case, this empirical probability should be $\sum_{i,j} w_{ij} P(k|i)P(k|j)$. For the second term in Eq. (6), $\frac{\mathcal{A}(V_k,V)}{\mathcal{A}(V,V)}$ is the empirical probability that a randomly selected edge is related to (i.e., has at least one end in) community k. For our case, this empirical probability should be $\sum_i P(k|i) \sum_j w_{ij}$. By summing these two terms over all k's and noticing that $P(k|i) = (D^{-1}X\Lambda)_{ik}$, we get formula Eq. (7). In addition, it is straightforward to verify that Q_s is equal to Q when $D^{-1}X\Lambda$ becomes a 0/1 indicator matrix. □