Co-Evolution of Multi-Typed Objects in Dynamic Star Networks

Yizhou Sun, Jie Tang, *Member, IEEE,* Jiawei Han, *Fellow, IEEE,* Cheng Chen, and Manish Gupta

Abstract—Mining network evolution has emerged as an intriguing research topic in many domains such as data mining, social networks, and machine learning. While a bulk of research has focused on mining the evolutionary pattern of homogeneous networks (e.g., networks of friends), however, most real-world networks are heterogeneous, containing objects of different types, such as authors, papers, venues, and terms in a bibliographic network. Modeling *co-evolution of multi-typed objects* can capture richer information than that on single-typed objects alone. For example, studying co-evolution of authors, venues, and terms in a bibliographic network can tell better the evolution of research areas than just examining co-author network or term network alone. In this paper, we study mining co-evolution of multi-typed objects in a special type of heterogeneous networks, called star networks, and examine how the multi-typed objects influence each other in the network evolution. A Hierarchical Dirichlet Process Mixture Model-based evolution model is proposed, which detects the co-evolution of multi-typed objects in the form of multi-typed cluster evolution in dynamic star networks. An efficient inference algorithm is provided to learn the proposed model. Experiments on several real networks (DBLP, Twitter, and Delicious) validate the effectiveness of the model and the scalability of the algorithm.

Index Terms—Information network analysis, data mining, co-evolution, clustering, dynamic star networks

1 Introduction

With the advent of social Web, social and information networks are ubiquitous. Examples include social networks (*e.g.*, MySpace, Facebook, Foursquare), microblogs (*e.g.*, Twitter, Jaiku), bibliographic databases (*e.g.*, DBLP, PubMed), and sharing systems (*e.g.*, Delicious, Flickr). Since networks are rather dynamic, it is interesting to study how objects in the networks form different clusters and how clusters evolve over time. The clusters represent groups of objects that are closely linked to each other, either due to hidden common interests or due to some social events. Most recent studies on cluster detection and evolution in networks [5], [11], [19], [6], [8] are on homogeneous networks, however, most real-world networks are heterogeneous, containing objects of different types, such as authors, papers, venues and terms in a bibliographic network like DBLP. One may wonder if modeling *co-evolution of multi-typed objects* may capture richer semantic information than that of single-typed objects. Let's examine an example.

- *Yizhou Sun is with the College of Computer and Information Science, Northeastern University, Boston, MA, 02115. E-mail: yzsun@ccs.neu.edu*
- *Jie Tang is with the Department of Computer Science and Technology, Tsinghua University, Beijing, China, 100084. E-mail: jietang@tsinghua.edu.cn*
- *Jiawei Han is with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 61801. E-mail: hanj@cs.uiuc.edu*
- *Cheng Chen is with MIT Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, Cambridge, MA, 02139. E-mail: chengch@mit.edu*
- *Manish Gupta is with Microsoft India (R&D) Private Limited, Hyderabad, India, 500032. E-mail: gmanish@microsoft.com*

Example 1. To study how research areas are evolving in the computer science bibliographic network formed by the DBLP data (hence called the *DBLP network*), one may model the evolution of author communities only (*e.g.*, author collaborations), or topics only (*e.g.*, hot terms in papers). However, by putting authors, papers, terms, and venues together, one may discover the dynamics of an area, such as *in the web mining community, not only new researchers joined in, but also they brought in fresh terminology and set up new venues and influenced many senior authors.* Such modeling can uncover the hidden *multi-typed clusters* at different time periods and their evolutionary structure, such as evolving, splitting, and merging in heterogeneous networks. ■

Fig. 1 shows a partial evolutionary structure of clusters from the DBLP network, where a rectangle box denotes a cluster with top occurred objects displayed and the area of the box denotes the relative size of the cluster (measured by the number of papers). It shows both the evolution of multiple types of objects in each cluster and the evolutionary structure among the clusters. For example, database and software engineering first formed a huge cluster, but later, database, data mining and machine learning merged into a big cluster, and software engineering itself became an independent cluster. As illustrated, each cluster is composed of objects from different types, and the cluster's evolution is determined by the co-evolution of objects from different types.

This problem is non-trivial, and it poses several challenges: (1) how to take different types of objects collectively to detect clusters? (2) how to discover the evolutionary structure (split, merge, and evolve) among clusters of different time windows by modeling the co-evolution of

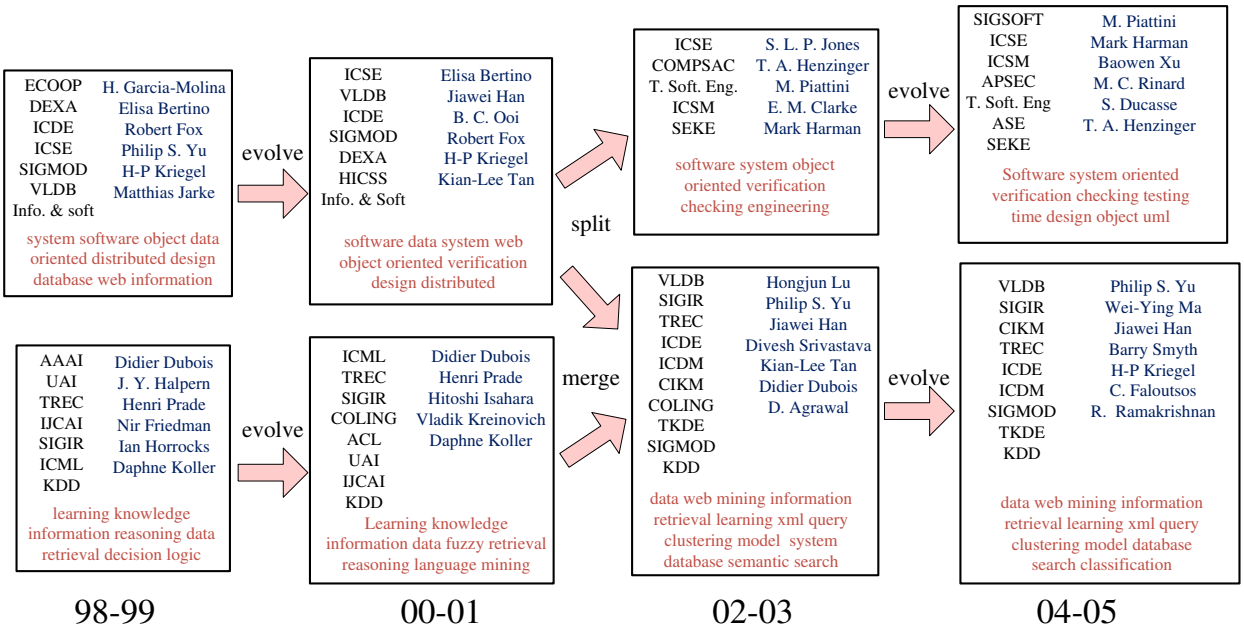

Fig. 1. An example of evolutionary clusters extracted from the DBLP data.

objects in each type? and (3) how to develop an efficient algorithm to solve the problem, as real-world information networks can be very large?

This paper develops a systematic approach to address the challenges for a particular type of heterogeneous networks, the *star networks*, which contain nodes from a center type and multiple attribute types, and links only exist between the center type and attribute types. Most event-based networks can be modeled into star networks: for the DBLP network, papers (as events) belong the center type, linking to authors, conferences, and terms as attribute types; and for the Twitter network, tweets can be taken as the center type, linking to users, text, hashtags, and so on. Then, a probabilistic model called *Evo-NetClus* is proposed to model the co-evolution of different types of objects. Both the content evolution (object distribution evolution in a cluster) and the evolutionary structure of the clusters (the dependency relationship between clusters) are modeled. An efficient algorithm is proposed to learn the model.

In all, the study makes the following contributions:

1) A novel problem of studying the co-evolution of multi-typed objects in dynamic heterogeneous star networks is proposed, with the goal of detecting evolutionary multi-typed clusters in such networks.
2) A Hierarchical Dirichlet Process Mixture Model-based generative model (*Evo-NetClus*) is proposed to solve the problem and an efficient algorithm is proposed for the model learning; and
3) Experimental results on several real datasets have demonstrated the effectiveness of the proposed model and the efficiency of the algorithm.

The remaining of the paper is organized as follows. Section 2 introduces preliminary concepts on heterogeneous information networks and Dirichlet process. Section 3 develops a generative model *Evo-NetClus* for evolutionary heterogeneous clusters in dynamic networks. Section 4 presents an efficient greedy learning algorithm. Section 5 reports the experiment results, Section 6 discusses the related work, and Section 7 concludes the study.

2 Preliminaries

In this section, we first introduce some preliminary knowledge and definitions, then formalize the evolutionary multi-typed cluster detection problem, and finally introduce the Dirichlet process that our model is built upon.

2.1 Heterogeneous Information Networks

A **heterogeneous information network** $\mathcal{G}$ is a network containing multiple types of objects and/or multiple types of links. Among all the heterogeneous networks, we consider in this paper only those with star network schema, i.e., links only exist between the center type of objects (*e.g.*, papers) as **target objects**, and several other types of objects (*e.g.*, authors, conferences, and terms) as **attribute objects**. Many of the heterogeneous networks can be modeled with the star network schema. For example, the DBLP bibliographic network can be modeled with research papers as the target objects, and their authors, terms used in the papers, and venues where these papers published as attribute objects; the Twitter network can be modeled with tweets as target objects, and users, terms and hashtags as attribute objects; and the Flickr network can be modeled with images as target objects and users, groups, and tags as attribute objects. A more detailed example of DBLP bibliographic information network with star network schema is introduced below.

Example 2 (Star bibliographic network.) A DBLP bibliographic network consists of information about research

papers, each *written* by a group of authors, *using* a set of terms, and *published* in a venue (a conference or journal). Such a bibliographic network is composed of four types of objects: *authors, venues, terms*, and *papers*. Links exist between papers and authors by the relationship of "write" and "written by," between papers and terms by "contain" and "contained in," between papers and venues by "publish" and "published by." ■

2.1.1 Multi-typed Cluster

Instead of the traditional clusters containing objects from one single type, we are interested in clusters containing multiple types of objects, namely, **multi-typed clusters** [23]. Given a heterogeneous network, a multi-typed cluster is a subnetwork of it, which contains different types of objects that (partially) belong to it. In this paper, we further define the **size of a cluster** as the number of target objects assigned to the cluster, and the **number of occurrences of an attribute object** in a cluster as the number of target objects linked to this object in the cluster. For example, we say the occurrence of an author is 4 in data mining cluster, if he/she has (published) 4 papers belonging to data mining area.

Following [23], we model each multi-typed cluster as a statistical model parameterized by distributions of attribute types (parameters denoted as ϕ), which defines the probability of target objects in each cluster according to their linked attribute objects. In this paper, we use the parameter vector ϕ to represent its corresponding multi-typed cluster model. Given the statistical model for each cluster, we can then assign the target objects into the cluster with the maximum posterior probability, and get the induced subnetwork of the cluster containing these target objects and their linked attribute objects accordingly.

2.1.2 Network Sequence

Objects and links in the networks are usually associated with time. A dynamic network can then be segmented into a **network sequence** according to such time information. We denote a network sequence as $\mathcal{GS} = (\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_T)$, where $\mathcal{G}_t = \langle V_t, E_t \rangle$ is the network associated with time window t. In the DBLP case, each $\mathcal{G}_t$ is a network comprised of all the papers published in year t or time window t, as well as the authors, venues and terms linking to them. In the Twitter case, each $\mathcal{G}_t$ is a network comprised of all the tweets posted in the time window t, and all the users, terms and hashtags associated with these tweets.

2.1.3 Co-Evolution Problem Formulation

In this paper, we consider the co-evolution problem in an *online mode*. That is, we try to detect the evolutionary multi-typed clusters at time $t+1$, once the network $\mathcal{G}_{t+1}$ is observed at time $t+1$. Although algorithms in *offline mode* that use all-time network information can produce a global view of evolution, online mode algorithm is more realistic for detecting evolution for real-time large-scale networks.

The evolutionary multi-typed cluster detection problem is then formalized as follows. Given an input network sequence $\mathcal{GS} = (\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_T)$, the output is a **$T$-partite graph**, where the nodes are the multi-typed clusters $\{\phi_{t,k}\}_{k=1}^{K_t}$ defined as statistical models for each network $\mathcal{G}_t$, and K_t is the automatically learned number of clusters for $\mathcal{G}_t$. A dependency link exists between two clusters in adjacent networks, say $\phi_{t-1,k'}$ and $\phi_{t,k}$, iff the cluster $\phi_{t,k}$ is (partially) evolved from $\phi_{t-1,k'}$, namely, there exists a target object that is assigned to cluster k in time window t and would be assigned to cluster k' in time window $t-1$ given the statistical models.

2.2 Dirichlet Process Mixture Model

We now first briefly introduce the Dirichlet process mixture model (DPM), which our evolution model is based on. Mixture model is a frequently used method in clustering, which assumes an observation o_i is generated from a fixed number, say K, of different statistical models $\{\phi_k\}_{k=1}^{K}$ (clusters), with different component weights π_k. A mixture model can be formalized as $o_i \sim \sum_{k=1}^{K} \pi_k p(o_i|z_i = k)$.

However, it is usually difficult for people to specify the correct cluster number K in the mixture model. Dirichlet process mixture model is a typical way to solve the problem, where the number of potential clusters is countable infinite, and the distribution of components follows a Dirichlet process (an extension of Dirichlet Distribution to infinite space) with a base distribution G_0. We follow the work [15] and define the DPM model as in Equation 3:

$$\begin{aligned} G &\sim \mathcal{DP}(\alpha, G_0) \\ \theta_i|G &\sim G \\ o_i|\theta_i &\sim f(\theta_i) \end{aligned} \tag{1}$$

where G is generated by a Dirichlet Process with the base measure G_0 and concentration parameter α, with the meaning of precision of G around base distribution G_0, θ_i follows the distribution of G, and o_i is generated from cluster with parameter θ_i.

According to [15], this model is equivalent to the following finite mixture model (Equation 2), when the cluster number K goes to infinity:

$$\begin{aligned} o_i|z_i, \phi_{1:K} &\sim f(\phi_{z_i}) \\ z_i|\pi &\sim Discrete(\pi_1, \ldots, \pi_K) \\ \phi_{1:K} &\sim G_0 \\ \pi &\sim Dirichlet(\alpha/K, \ldots, \alpha/K) \end{aligned} \tag{2}$$

where z_i stands for the latent class label of the observation o_i. In this model, given the cluster number K, the parameters $\phi_{1:K}$ for all the clusters are drawn from the same prior distribution G_0, and the component weights $\pi_{1:K}$ are drawn from a Dirichlet distribution as the prior.

Later, researchers extended DPM into hierarchical Dirichlet process mixture model (HDP) [26], which can generate clusters for objects coming from different groups, where different groups may share some common clusters across groups but also have some unique clusters within

the group. Formally,

$$\begin{aligned} G_0 &\sim \mathcal{DP}(\gamma, H) \\ G_j &\sim \mathcal{DP}(\alpha, G_0) \\ \theta_{ji}|G_j &\sim G_j \\ o_{ji}|\theta_{ji} &\sim f(\theta_{ji}) \end{aligned} \tag{3}$$

where γ is the first level concentration parameter, H is the first level base distribution, G_j is the set of measures for group j, and θ_{ji} is the component parameter for o_{ji}, which is drawn from G_j.

3 Modeling Evolutionary Multi-Typed Clusters

In this section, we propose a probabilistic model to solve the evolutionary multi-typed cluster detection problem.

3.1 Overview

We decompose the problem of modeling evolutionary clusters into two sub-problems: (i) model multi-typed clusters with flexible numbers for static networks, and (ii) model the evolutionary structure among them. A probabilistic model *Evo-NetClus* that solves the two sub-problems in a unified framework is proposed.

First, we propose the *DP-NetClus* model, which is able to detect multi-typed clusters with flexible numbers using the objects from multiple types collectively in a static network. To automatically detect the best number of clusters is important, as it is unrealistic for human to specify the number of clusters in every time window, considering the emergence of new clusters and the fading-out of old ones.

Second, in order to model the evolutionary structure between clusters of adjacent time windows, we propose *Evo-NetClus* on top of *DP-NetClus*. That is, the evolutionary network clustering is an generative model for target objects at current time window, by using both the prior knowledge of clusters learned from the previous time window and the background knowledge in the current time window.

To better illustrate the idea of evolutionary cluster detection and probabilistic modeling, we take the DBLP bibliographic network as an example. Nevertheless, the model and algorithm can be used in any heterogeneous networks with a star network schema.

3.2 DP-NetClus: Modeling Multi-Typed Clusters

Similar to the ranking-based clustering algorithm for star networks, *NetClus* [23], the probability of a target object in a multi-typed cluster is defined as the conditional probability in that cluster, given their linked attribute objects. In addition, we model two more factors. First, in *NetClus*, the number of the clusters is specified by users, which is a difficult task in a dynamic scenario. Thus we add a prior to the component coefficients to guide the selection of the number of clusters. Second, we add a prior to the object distributions in the model, which makes the whole process as a generative model. The two priors can be unified in one framework by using the Dirichlet process [15]. We call our new multi-typed network clustering model as *DP-NetClus*, as it is a Dirichlet process-based network clustering model.

Specifically, in the DBLP network, there are four types of objects: papers (O), authors (A), venues (C), and terms (W), where links exist only between O and each of the other three types. For each target object (paper) $o_i \in O$, it is represented as a tuple $(\mathbf{a}_i, \mathbf{c}_i, \mathbf{w}_i)$, where $\mathbf{a}_i = (n_{a_{i1}}, \ldots, n_{a_{i|A|}})$ is a vector with length of $|A|$, $|A|$ denotes the size of the author set, and $n_{a_{ij}}$ denotes the weight of the link between o_i and author $a_j \in A$. $\mathbf{c}_i$ and $\mathbf{w}_i$ have similar meanings. In the *author* vector, $n_{a_{ij}}$ is a binary value, denoting whether author a_j has written paper o_i or not; in the *venue* vector, $n_{c_{ij}}$ is a binary value, denoting whether o_i is published in venue c_j; and in the *term* vector, $n_{w_{ij}}$ is a positive integer indicating the number of times o_i has used term w_j. We model each target object o_i generated from a mixture model: $p(o_i) = \sum_k \pi_k p(o_i|k)$, where π_k is the component probability for cluster k.

We assume that for every attribute type, the objects within that type follow a discrete distribution in each cluster k. Each target object o_i in the cluster k is modeled as a joint probability of its linked attribute objects in k. Let ϕ_k be the parameter vector for the statistical model for cluster k, which can be further decomposed into three components, i.e., $\phi_k = (\phi_k^A, \phi_k^C, \phi_k^W)$, where ϕ_k^A is the parameter vector for the conditional discrete distribution of Type A in cluster k, i.e., $\phi_k^A(a)$ denotes the probability of author a in cluster k within the author type (similarly for ϕ_k^C and ϕ_k^W). The probability of generating paper o_i in cluster k is then $p(o_i|k) = p(\mathbf{a}_i|k)p(\mathbf{c}_i|k)p(\mathbf{w}_i|k)$, where $p(\mathbf{a}_i|k) = \prod_{j=1}^{|A|}(\phi_k^A(a))^{n_{a_{ij}}}$, according to the discrete distribution, and similarly for $p(\mathbf{c}_i|k)$ and $p(\mathbf{w}_i|k)$.

In order to model the flexible number of clusters and add prior knowledge to distributions for attribute types, we introduce Dirichlet process (DP) [25] as the prior for the mixture model, which could have infinite number of clusters. In other words, cluster parameters are assumed to be generated from G, which is generated from a Dirichlet process, $G \sim DP(\alpha, G_0)$, where α is a concentration parameter, which controls the probability of creating a new cluster, and G_0 is the base distribution for each cluster model. Notice that, Dirichlet process is a distribution over distributions, which defines the distribution for all the possible cluster models. In our model, the base distribution G_0 is defined as three independent symmetric Dirichlet distributions for the three attribute types, with concentration parameters as β_A, β_C and β_W (denoted as $\boldsymbol{\beta}$ collectively).

The DP-based generative model for target objects in networks can be described in the following generative process (Figure 2):

- Sample cluster component coefficients $\pi = \{\pi_k\}_{k=1}^{\infty}$. $\pi|\alpha \sim GEM(\alpha)$ (GEM denotes the stick-breaking processing [25]), namely, $\pi'_k|\alpha \sim Beta(1, \alpha)$, $\pi_k = \pi'_k \prod_{l=1}^{k-1}(1 - \pi'_l)$.
- Sample cluster component models $\{\phi_k\}_{k=1}^{\infty}$. $\phi_k^A \sim H_A(\beta_A)$, $\phi_k^C \sim H_C(\beta_C)$, and $\phi_k^W \sim H_W(\beta_W)$,

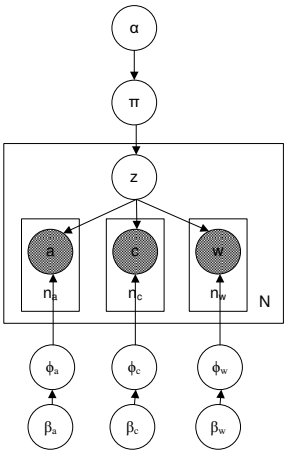

Fig. 2. Graphical Model for *DP-NetClus*.

where H_A, H_C, and H_W are background distributions (prior distributions) for each type of attribute objects. Particularly, we assume all ϕ_k's are multinomial distributions and the prior distributions H's are symmetric Dirichlet distribution, namely H_A is a $|A|$-dimensional Dirichlet distributions, i.e., $Dir(\beta_A, \ldots, \beta_A)$. H_C and H_W are similarly defined.

- Sample papers. For each paper, first sample its cluster z according to the component coefficients, $z \sim \pi$; then for each type of attribute objects, sample authors, conferences, and terms according to the cluster models, with the probability of $\phi_z^A(a)$, $\phi_z^C(c)$, and $\phi_z^W(w)$. In other words, $o|z \sim F(\phi_z)$, i.e., object o in cluster z follows the distribution of cluster model with parameter ϕ_z.

To sum up, the probability for generating all the target objects in the network under the hyper-parameters α and $\boldsymbol{\beta}$ is then:

$$p(\mathbf{o}|\alpha, \boldsymbol{\beta}) = \int_{\boldsymbol{\pi}} p(\boldsymbol{\pi}|\alpha) \prod_i \sum_{k=1}^{\infty} \pi_k p(\mathbf{a}_i|k, \beta_A) p(\mathbf{c}_i|k, \beta_C) p(\mathbf{w}_i|k, \beta_W)$$

where $p(\mathbf{a}_i|k, \beta_A)$ is the probability of generating the author set for paper o_i in cluster k under hyper-parameter β_A, which is calculated by $\int_{\phi_k^A} p(\mathbf{a}_i|\phi_k^A) p(\phi_k^A|\beta_A)$, where $\phi_k^A|\beta_A \sim Dir(\beta_A, \ldots, \beta_A)$, the symmetric Dirichlet distribution. The formulas are similarly defined for other attribute types.

3.3 Evo-NetClus: Modeling Evolutionary Clusters

In order to model the evolution of clusters, we need to model the evolution structure between clusters in addition to the clusters themselves. We then propose the hierarchical Dirichlet process-based [26] model *Evo-NetClus*, which is built on top of *DP-NetClus*, to solve the problem.

As the network sequence comes in a streaming way in the real world, we model the evolution of clusters in an incremental (online) way: a cluster in time window t is only dependent on the clusters of previous time window $t-1$ and the network in current time window t. Notice that, the desired cluster models at time window t denoted as $\{\phi_{t,k}\}_{k=1}^{K_t}$ are a result of *co-evolution* of different types of attribute objects from the cluster models of the previous time window $t-1$ denoted as $\{\phi_{t-1,k'}\}_{k'=1}^{K_{t-1}}$.

In order to model the evolution structure, intuitively, we first assume all the target objects at time t are generated from clusters at time $t-1$, then they are re-structured into new clusters, which represent a better hidden structure of the networks. These new clusters are generated in a way that they can capture both the common clusters across different old clusters and the special clusters within single old clusters. By using hierarchical Dirichlet process, we can model the dependency relationship between the old clusters (at time $t-1$) and the new clusters (at time t), and thus the evolution structure.

Given the clusters learned at time $t-1$, denoted as $\{\phi_{t-1,k'}\}_{k'=1}^{K_{t-1}}$, they are treated as prior knowledge to guide the formation of clusters in time t. First, we use the learned cluster models at time $t-1$ to partition target objects into groups, each representing a "pseudo" cluster as if the cluster models do not change. We group target objects at time window t into $K_{t-1}+1$ prior groups according to the posterior probability it is generated from each historical cluster model and a new cluster under background model H_t. The calculation of these posteriors will be introduced in Section 4. Note that, these prior groups would be exactly the clustering results for time window t if the cluster models do not change at all. Then, to model the dependency between previous clusters and current clusters, we model objects in each prior group j as a mixture model of new cluster models $\{\phi_{t,k}\}_{k=1}^{K_t}$, which follows a hierarchical Dirichlet process (HDP) $G_{t,j} \sim DP(\alpha, G_{t,0})$, where $G_{t,0}$ is a higher level Dirichlet process $G_{t,0} \sim DP(\gamma, H_t)$. H_t is the symmetric dirichlet distribution for each type of objects at time t, and γ is the higher level concentration parameter. Hence, mixture models for objects in different prior groups share the same cluster models (sampled from $G_{t,0}$) but with different proportions (dependent on γ and α).

Second, we use the previous cluster models at time $t-1$ as prior distributions for the current cluster models. In other words, we consider objects at time $t-1$ with their cluster assignments as prior observations of cluster models at time t, but with a damping factor to discount their impacts. For example, we treat a paper o_i with cluster assignment k at time $t-1$ as a paper observed in cluster k at time t, but it is only counted as a much smaller ratio. The *posterior DP* under these prior clusters for time t is then:

$$\begin{aligned} & G_{t,0}|\boldsymbol{\eta}_{t-1}, \{\phi_{t-1,k'}\}, \lambda, H_t \\ \sim\ & DP(\gamma+\lambda, \frac{\gamma H_t + \lambda \sum_{k'=1}^{K_{t-1}} \eta_{t-1,k'} \delta\phi_{t-1,k'}}{\gamma+\lambda}) \end{aligned} \quad (4)$$

where γ is the concentration parameter for the higher level DP, and λ denotes the total impact coming from clusters at time $t-1$ (the strength of pre-existing clusters); $\eta_{t-1,k'}$ denotes the proportion of the size of each cluster k' at time $t-1$ and $\{\phi_{t-1,k'}\}$ denotes the statistical models of clusters at time $t-1$. Notice that, if the number of target objects at time $t-1$ is N_{t-1}, through simple calculation it could be found that each target object at time $t-1$ is damped

with a rate $\frac{\lambda}{N_{t-1}}$. By re-examining the base distribution in the posterior DP, one can see that the prior distributions for new clusters are determined by a mixture model of the previous cluster distributions at time $t-1$ and the current background distribution H_t. We can also see that, the evolution of clusters are determined by the evolutions of objects in each type collectively, since each cluster model is determined by distributions of all the attribute types.

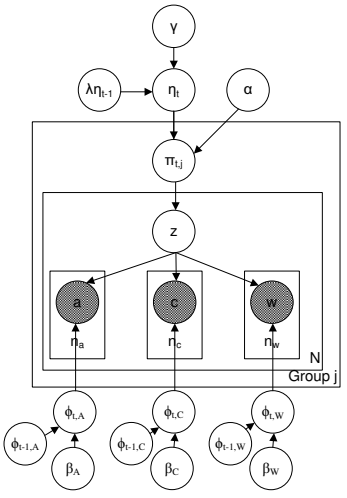

Fig. 3. Graphical Model for *Evo-NetClus*.

The generative process for *Evo-NetClus*, given the knowledge of the prior group for each object at time window t, is described as follows, with the graphical representation of *Evo-NetClus* shown in Figure 3:

- Sample global cluster component coefficients: $\eta_t \sim GEM(\gamma+\lambda)$, where GEM denotes the stick-breaking processing as introduced in *DP-NetClus*.
- Sample group-wise cluster component coefficients $\pi_{t,j}$, where $\pi_{t,j}|\alpha,\eta_t \sim DP(\alpha,\eta_t)$, namely within group j, component coefficients are sampled from the higher level component coefficients distribution η_t, with a concentration parameter α.
- Sample cluster component models: $\phi_{t,k} \sim \frac{\gamma H_t+\lambda\sum_{k'=1}^{K_{t-1}}\eta_{t-1,k'}\delta_{\phi_{t-1,k'}}}{\gamma+\lambda}$. That is, for each cluster k, it is either sampled from background distribution H_t with probability $\frac{\gamma}{\gamma+\lambda}$, or sampled from an existing cluster k' at time $t-1$ with probability $\frac{\lambda\eta_{t-1,k'}}{\gamma+\lambda}$.
- Sample papers o_{ji} for each group j: first sample its cluster label according to $z_{ji}\sim\pi_{t,j}$, and then sample its attribute objects according to $o_{ji}\sim F(\phi_{t,z_{ji}})$.

Notice that, $\pi_{t,j,k}$, the kth item in group-wise component coefficient vector $\pi_{t,j}$, is the proportion that target objects belong to cluster j at time $t-1$ but belong to cluster k at time t. From this, we can track where a cluster is evolving to. A T-partite graph between clusters from adjacent time windows can thus be constructed.

Intuitively, HDP first clusters target objects in each prior group into micro clusters, and then clusters these micro clusters into macro clusters that are shared across different groups. Please refer to [26] for a more detailed explanation for this intuition. Thus, we can build the relationship between clusters from previous time and clusters of current time, as a prior group can be viewed as a linear combination of current clusters.

3.4 Discussions

We now provide some discussions on *Evo-NetClus*, about the issues on hyper-parameter settings and the potential extension to more general heterogeneous information networks.

3.4.1 Determine Hyper-Parameters in the Model

In the *Evo-NetClus* model, three sets of hyper-parameters are used: (i) γ and α, which are used to control the probability to generate a new cluster; (ii) $\boldsymbol{\beta}=(\beta_A,\beta_C,\beta_W)$, associated with Dirichlet prior distributions for each type of attribute objects; and (iii) λ, which controls the smoothness between clusters in adjacent time windows: the larger λ the larger the weight for a previous target object in the current time window.

By using different settings for the hyper-parameters γ, α, and $\boldsymbol{\beta}$, we can control the granularity of detected clusters. For one level DP with parameter α and fixing $\boldsymbol{\beta}$, the expected number of clusters are $O(\alpha\log(1+\frac{n}{\alpha}))$ [25], where n is the number of target objects. $\boldsymbol{\beta}$ can be considered as the smoothing parameter as introduced in [31] for the Dirichlet smoothing method. For a larger $\boldsymbol{\beta}$, it means more smoothing on the distributions for attribute objects, and therefore, the larger chance a target object goes to an existing cluster than a new one, and thus the fewer clusters. For the smoothing parameter λ, we have shown in the experiment section that, a good level of smoothness can improve the quality of detected clusters.

Another question is how to determine these hyper-parameters. A frequently used method is to add further hyper-parameters, for example, the gamma distribution for γ and α as used in [32], and these hyper-parameters can be estimated by optimizing the model. In practice, these values can be set empirically according to tentative runnings of the algorithm, and they are dependent on the granularity of the clusters required by the users.

Note that, these hyper-parameters only need to be set once for the whole network sequence, and the model controls the granularity level of the clusters and automatically determines whether to create a new cluster, split an existing cluster, an so on. This provides more practical usability than evolutionary algorithms that need to specify number of clusters for each time window.

3.4.2 Modeling More General Heterogeneous Information Networks

In this paper, we have introduced the evolution model for a special type of heterogeneous information networks, which are star networks. We now give some discussion on how to model more general heterogeneous information networks.

More General Bibliographic Networks. In many bibliographic networks, there could be other link types, for

example, the citation links between papers. In this case, the papers can be treated as both target objects and attribute objects, such as in Link-LDA [14]. In addition to the the ranking distributions for venues, authors, and terms, we also need to model the ranking distribution for papers in terms of the probabilities of being cited in a certain research area. We then can define the generative probability of a paper by considering citation links in addition, and the evolution model can be extended straightforwardly.

In other cases, people may want to directly model the co-authorship between two authors. A straightforward way is to model the probability of the co-occurrences of author pairs in addition to the probability of the occurrence of single authors. Then the probability of a paper in a cluster is determined also by the probability of all pairs of authors for this paper in that cluster. By directly modeling co-authorship and other relationships, the clustering model can be further enriched due to the introduction of more parameters, but this is likely to cause overfitting as well.

Arbitrary Heterogeneous Information Networks. For information networks with arbitrary network structure, the major difficulty of directly applying current model lies in the difficulty of identifying *target objects*. A possible way to handle this issue is to treat each link as a virtual target object, and model the generation of links from different types separately. The ranking distributions of objects from different types are also modeled separately, but share the same value for the same type of objects, even they could be in different relations.

4 The Learning Algorithm for Evo-NetClus

Learning the *Evo-NetClus* model is to estimate the unknown parameters in the model. The main unknown parameters are distributions for all attribute types in different clusters $\{\mathbf{\Phi}_t\} = (\phi_t^A, \phi_t^C, \phi_t^W)$ in all time windows. Also, we need to learn the hidden cluster labels for each target object. It is usually intractable to perform exact inference in such a probabilistic model, and the typical method is to conduct approximate inference such as Gibbs sampling [15]. However, the Gibbs sampling algorithm needs to take thousands of iterations to converge, which is very time consuming, especially for large data sets.

In this paper, we propose an efficient greedy algorithm to learn *Evo-NetClus*. The general idea is that instead of assigning each target object to clusters following the conditional probabilities as in collapsed Gibbs sampling for HDP [26], we greedily assign each target object to the cluster with the highest conditional probability. This is similar to the idea used in k-means which always assigns a target object to the cluster that is with the closest distance, except that in our case we need to dynamically create new clusters and remove empty ones as no specified number of clusters is given.

In Section 4.1, we first introduce how to calculate the conditional probability of a target object in each potential cluster, and then we use it in Section 4.2 for the learning algorithm.

4.1 Calculate the Posterior Conditional Probability

First, we introduce how to calculate the conditional probability of a target object o_i generated from an existing cluster k or a new cluster k_{new} in a general case, given the assignment of rest of the objects, under a conjugate Dirichlet prior $H = (H_A, H_C, H_W)$, which will be used substantially in our learning algorithm.

The probability of a target object o_i generated from an existing cluster k given the rest of objects in k, $f_k^{-o_i}(o_i)$, can be calculated as the product of conditional probabilities of its linked attribute objects in each type:

$$f_k^{-o_i}(o_i) = f_k^{-\mathbf{a}_i}(\mathbf{a}_i) f_k^{-\mathbf{c}_i}(\mathbf{c}_i) f_k^{-\mathbf{w}_i}(\mathbf{w}_i)$$

where $f_k^{-\mathbf{a}_i}(\mathbf{a}_i)$ is the probability of observing authors $\mathbf{a}_i$ in cluster k given the rest of the authors. $f_k^{-\mathbf{a}_i}(\mathbf{a}_i) = \frac{\Gamma(\beta_A|A|+n_{k,A}^{-i})}{\Gamma(\beta_A|A|+n_{a_i}+n_{k,A}^{-i})} \frac{\prod_{j=1}^{|A|}\Gamma(\beta_A+n_{a_{ij}}+n_{k,A}^{-i}(j))}{\prod_{j=1}^{|A|}\Gamma(\beta_A+n_{k,A}^{-i}(j))}$, where n_{a_i} is the number of authors for paper o_i; $n_{a_{ij}}$ indicates whether a_j writes paper o_i; $n_{k,A}^{-i}$ is the total occurrences of authors in cluster k without considering paper o_i, and $n_{k,A}^{-i}(j)$ is the occurrence number of author a_j in cluster k without considering o_i. This can be calculated as the integral over posterior distributions under prior H_A, which is still a Dirichlet distribution with the updated parameters by the observed data. $f_k^{-\mathbf{c}_i}(\mathbf{c}_i)$ and $f_k^{-\mathbf{w}_i}(\mathbf{w}_i)$ can be similarly calculated.

The probability of object o_i generated from a new cluster can be directly calculated according to the integral over the prior distributions for each type, which are Dirichlet distributions:

$$f_{k_{new}}(o_i) = f_{k_{new}}(\mathbf{a}_i) f_{k_{new}}(\mathbf{c}_i) f_{k_{new}}(\mathbf{w}_i)$$

with $f_{k_{new}}(\mathbf{a}_i) = \frac{\Gamma(\beta_A|A|)}{\Gamma(\beta_A|A|+n_{a_i})} \frac{\prod_{j=1}^{|A|}\Gamma(\beta_A+n_{a_{ij}})}{\prod_{j=1}^{|A|}\Gamma(\beta_A)}$. Similarly, we can define $f_{k_{new}}(\mathbf{c}_i)$ and $f_{k_{new}}(\mathbf{w}_i)$.

4.2 The Learning Algorithm

Before we learn the algorithm at each new time window t, we need to first generate prior groups for all the target objects in t. The learning algorithm is an iterative algorithm, given the prior groups. At each iteration, each target object $(o_{t,i})$ is assigned to an existing cluster or a new cluster according to their maximum posterior probabilities in each cluster. Once a target object has changed its cluster, the statistics associated with the related clusters are dynamically updated. Finally, the algorithm stops when the cluster membership for target objects is not significantly changed. The algorithm is summarized in Algorithm 1, and the details are in the following three steps.

Step 0: Generating Prior Groups. First, we assign the target objects $o_{t,i}$ into different prior groups according to the posterior probabilities they belong to each cluster model. Here a prior group can represent an existing cluster at time $t-1$, or a new possible cluster at time t. Notice that, for the first time window $t = 1$, as there are no exiting clusters, all the objects go to one prior group. The posterior probabilities of a target object in different clusters

are determined by both the size of the cluster and the conditional probability of the target object in that cluster:

$$p(z_{t,i} = k|o_{t,i}, \mathbf{z}_{t-1}, \mathbf{O}_{t-1}, H_t) \propto n_{t-1,k} f_k^{-o_{t,i}}(o_{t,i})$$
$$p(z_{t,i} = k_{new}|o_{t,i}, H_t) \propto \gamma f_{k_{new}}(o_{t,i})$$

We then denote $o_{t,ji}$ as the i_{th} object in prior group j, which is simplified as o_{ji} when there is no ambiguity. For other notations, we also omit the time index t when there is no ambiguity, *e.g.*, η is short for η_t, and so on. Note that for background distribution H_t, we use the same $\boldsymbol{\beta}$ parameters for all time windows, but use a different attribute set, which is a union of attribute objects appeared in time $t-1$ and time t, as we need to consider the networks from both time windows.

Step 1: Iterative Hidden Cluster Label Assignment. Second, we follow the idea of collapsed Gibbs sampling [26], where the parameters for each cluster have been integrated out, but assign hard cluster labels to target objects. At each iteration, the target objects are randomly shuffled for a new order. For each object o_{ji} at an iteration, the target object is first greedily assigned to either an existing cluster or a new cluster, according to the maximum posterior probability; second, the statistics associated with each cluster are immediately updated if the object has changed its assignment. The two steps are introduced in the following.

1.1 Cluster Assignment Step: Assign object o_{ji} into an existing cluster k or a new cluster k_{new} according to the maximum posterior probabilities:

$$p(z_{ji} = k, k \notin \{K_{t-1}\}|\mathbf{z}_{-ji}, \mathbf{o}) \propto (n_{j\cdot k}^{-ji} + \alpha\eta_k) f_k^{-o_{ji}}(o_{ji})$$
$$p(z_{ji} = k, k \in \{K_{t-1}\}|\mathbf{z}_{-ji}, \mathbf{o}) \propto (n_{j\cdot k}^{-ji} + \lambda\frac{n_{t-1,k}}{N_{t-1}} + \alpha\eta_k) f_k^{-o_{ji}}(o_{ji})$$
$$p(z_{ji} = k_{new}|o_{ji}) \propto \alpha\eta_u f_{k_{new}}(o_{ij})$$

where $n_{j\cdot k}^{-ji}$ denotes the number of objects in group j belonging to cluster k without considering object o_{ji}, η_k is the higher level proportion coefficient for cluster k, and η_u is the proportion coefficient for unseen clusters. As the group-wise cluster coefficients are sampled from $DP(\alpha, \eta)$, the posterior probability naturally takes $\alpha\eta_k$ as pseudo counts for the size of cluster k in group j. Also, if k is an existing cluster at time $t-1$, i.e., $k \in \{K_{t-1}\}$, the objects in $t-1$ need to be considered with discount $\frac{\lambda}{N_{t-1}}$.

1.2 Statistics Adjustment Step: Once the object o_{ji} has changed its cluster label from k_{old} to k, several statistics need to be updated immediately for further calculating the conditional probability of a target object in the updated clusters.

1) Update m_{jk} and $m_{jk_{old}}$, which are the number of micro clusters in group j that belong to clusters k and k_{old}, respectively. For cluster k, this problem is equivalent to the question of "how many clusters are there for $n_{j\cdot k}$ objects following a DP process with concentration parameter $\alpha\eta_k$." For our greedy algorithm, we set it as its approximated expected value: $m_{jk} \simeq \alpha\eta_k \log(1 + \frac{n_{j\cdot k}}{\alpha\eta_k})$ [25]. $m_{jk_{old}}$ is similarly calculated.

Input: Network G_t, G_{t-1}, $\mathbf{z}_{t-1}$; γ, α, $\boldsymbol{\beta}$, λ ;
Output: The cluster assignment vector $\mathbf{z}_t$; the parameters $\boldsymbol{\Phi}_t = \{\phi_t^A, \phi_t^C, \phi_t^W\}$;

Assign each object into prior groups;
repeat
  for *each object* o_{ji} **do**
    1. Cluster Assignment Step: Assign o_{ji} to the cluster with the maximum posterior probability in either existing cluster k or a new cluster $k+1$;
    2. Statistics Adjustment Step: Update relevant statistics;
    3. if cluster k_{old} for o_i contains no objects, remove the cluster;
  end
until *reaches cluster change threshold*;
Estimate parameters $\boldsymbol{\Phi}_t$ for each cluster;

Algorithm 1: Parameter Estimation Algorithm.

2) Update the cluster proportion coefficients η. Conditioning on the assignment of target objects into clusters and micro clusters, according to [26], $(\eta_1, \eta_2, \ldots \eta_K, \eta_u) \sim Dir(m_{\cdot 1}, m_{\cdot 2}, \ldots, m_{\cdot K}, \gamma)$, where $m_{\cdot k}$ denotes the number of micro clusters associating with cluster k. For our greedy algorithm, we set η as the expected value, $\eta_k = \frac{m_{\cdot k}}{\sum_{k=1}^{K} m_{\cdot k} + \gamma}$, and $\eta_u = \frac{\gamma}{\sum_{k=1}^{K} m_{\cdot k} + \gamma}$.
3) Update the counting information, namely $n_{k,A}$, $n_{k,C}$, $n_{k,W}$, etc., for each type of objects in clusters k and k_{old}.

Notice that, if cluster k_{old} becomes empty after the re-assignment, it needs to be removed from the cluster list.

Step 2: Cluster Distribution Estimation. Once the assignment for each object is fixed, the parameter ϕ_k for each cluster can be estimated accordingly. Specifically, the parameter for each component $\phi_k = (\phi_k^A, \phi_k^C, \phi_k^W)$ is a Dirichlet distribution given the observations in cluster k, and has the MAP estimation as:

$$\phi_k^A(j) = \frac{\beta_A + n_j^A}{\beta_A|A| + n^A}; \phi_k^C(j) = \frac{\beta_C + n_j^C}{\beta_C|C| + n^C}; \phi_k^W(j) = \frac{\beta_W + n_j^W}{\beta_W|W| + n^W}.$$

4.3 Time Complexity Analysis

It is easy to see that for each iteration of Step 1 of the learning algorithm, which contributes the major computation cost, the time complexity of the learning algorithm for each time window is $O(dn \log n)$, where n is the number of target objects in that time window, d is the average number of attribute objects linked to each target object in the network, and $\log n$ is the average number of clusters in that time window.

5 EXPERIMENT

Here we study the effectiveness and efficiency of the *Evo-NetClus* model and algorithm using real datasets.

5.1 Datasets

The experiments were performed on three real datasets.

1) The 20-year DBLP network sequence from year 1990 to 2009, which contains four types of objects: papers

($645K$), authors ($50K$), conferences (journals) ($1K$) and terms (10K). Terms are extracted from paper titles. Stop words and low frequency objects have been removed.

2) The 8-week Twitter network sequence from March 1 to May 29 of 2011, which is a sample (1%) of all the tweets via Twitter API. The network contains four types of objects: tweets ($1994K$), the users who posted more than 5 tweets ($68K$), the terms appeared more than 35 times ($30K$), and the hashtags appeared more than 10 times ($12K$).
3) The half-year (26-week) Delicious network sequence from July. 1 to Dec. 29, 2010. The network also contains four types of objects, namely tagging events ($330K$), users ($63K$), web URLs ($102K$) and tags ($58K$), following a star network schema, where tagging events are the target objects. Objects that are with low frequency have been removed.

5.2 Effectiveness Study

To evaluate the effectiveness of our model, a subset of DBLP called "four-areas" dataset is used, which includes 22 major conferences[1] in data mining related areas such as data mining, information retrieval, database, and machine learning. We set each time window with the length of 4 years, and thus obtain a network sequence with 5 segments. For the hyper parameters, we set $\gamma = 1, \alpha = 1, \lambda = 10, \beta_A = \beta_C = 0.05, \beta_W = 0.1$ by default.

5.2.1 Quality of Clusters

To evaluate the quality of clustering, we use Normalized Mutual Information (NMI) [21] to compare the clusters detected by *Evo-NetClus* and three baselines.

The three baselines are (1) *NetClus* [23], which is a static multi-typed clustering algorithm for heterogeneous networks and is used to demonstrate the power of our evolution model over static clustering models; (2) a spectral clustering algorithm on multi-mode networks proposed in [24], which we only use the static clustering part, due to its difficulty in handling variant object sets, and is used to demonstrate the advantages of our probabilistic approach-based model in dealing with networks that are very sparse and with variant object sets; and (3) *Evo-Text*, which is *Evo-NetClus* limited on text type and is used to demonstrate the power of co-evolution over evolution on a single attribute type. We test the clustering results for conference type according to the four-area ground truth, where each conference is labeled as one of the four areas. For *Evo-Text* and *Evo-NetClus*, the algorithms are applied on the whole network sequence, with no input of the number of clusters. *NetClus* and spectral clustering are only applied on G_5, and the input number of clusters are set as 8, which is learned by *Evo-NetClus*. We then apply the k-means further by setting $K = 4$ to get the final 4 clusters for conferences according to either the soft clustering results or spectral vectors, and then use NMI to compare the hard clustering results with the ground truth. The results are shown in Fig. 4, which is consistent with our assumptions.

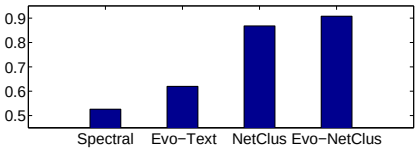

Fig. 4. Comparison of Clustering Accuracy for Venues in NMI)

In addition, we compare *Evo-NetClus* with *Evo-Text* and recurrent CRP-based evolution algorithms that are extended from [2], using measures including precision, recall, and F-measure, where whether a pair of objects in the output cluster truly belongs to the same cluster in the ground truth is evaluated. The results are summarized in Table 1. *RCRP-All* denotes the recurrent CRP evolution algorithm using all types of objects, and *RCRP-Text* denotes the evolution algorithm using text information only. We can see that *Evo-NetClus* has the best F-measure, compared with other evolution algorithms that either considers only partial attribute information or models simpler evolution structure.

TABLE 1
Comparison of Clustering Accuracy for Venues in Precision, Recall, and F-Measure

	Precision	Recall	F-Measure
Evo-NetClus	**1.0000**	0.6600	**0.7952**
Evo-Text	0.3082	**0.9000**	0.4592
RCRP-All	**1.0000**	0.5800	0.7342
RCRP-Text	0.2647	0.5400	0.3553

5.2.2 More Analysis on Co-Evolution

To further evaluate the quality of co-evolution, we propose two new measures and test them on the "four-areas" dataset.

The first measure evaluates how well we can model each target object's cluster membership by studying their posterior membership vectors, called *cluster compactness*. Intuitively, the more similar two target objects, the more similar their posterior cluster probabilities. In DBLP, two papers published in the same conference should be usually more similar than those published in different conferences. Thus, we define cluster compactness for papers as the ratio between the average similarity score within the same conference and the average similarity score between different conferences:

$$Compactness = \frac{\sum_{i\neq j} \cos(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j)\mathbf{1}_{\{c(i)=c(j)\}}}{\sum_{i\neq j} \mathbf{1}_{\{c(i)=c(j)\}}} \Big/ \frac{\sum_{i\neq j} \cos(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j)\mathbf{1}_{\{c(i)\neq c(j)\}}}{\sum_{i\neq j} \mathbf{1}_{\{c(i)\neq c(j)\}}} \quad (5)$$

where $\boldsymbol{\omega}_i$ is the K dimensional posterior cluster membership vector, $\omega_{ik} \propto n_k f_k^{-o_i}(o_i)$ denotes the posterior probability target object o_i belonging to cluster k, and K is the learned number of clusters; $c(i)$ and $c(j)$ denotes the

1. KDD, ICDM, SDM, PKDD, PAKDD, SIGIR, ECIR, TREC, ACL, CIKM, WSDM, WWW, NIPS, ICML, UAI, IJCAI, AAAI, SIGMOD, VLDB, ICDE, PODS, EDBT.

conferences for papers o_i and o_j, and $\mathbf{1}$ is the indicator function, which is 1 if the predicate is true otherwise 0. We use cosine to measure the similarity between two membership vectors. Note that, the higher compactness, the higher quality of the clusters.

The second measure is to evaluate how well we can separate different clusters by studying the differences of the conditional distributions for each type of attribute objects in different clusters, called *average KL-divergence*. Intuitively, the better quality of the cluster partition for the network, the more scattered the conditional distributions of attribute types in different clusters. In DBLP, we expect that for two different clusters, the author, conference, and term distributions should be quite distinct from each other. We then use average KL-divergence to quantify the pair-wised conditional distribution differences for each attribute type:

$$avgKL_A = \frac{\sum_{k_1 \neq k_2} KL(\phi^A_{k_1} || \phi^A_{k_2})}{K(K-1)} \quad (6)$$

where ϕ^A_k is the parameter vector for the discrete distribution of the author type, K is the final number of clusters. $avgKL_C$ and $avgKL_W$ are similarly defined for conference and term types. To measure the overall average KL-divergence over all the attribute types, we use $avgKL_{all}$, the mean of the average KL-divergence for each attribute type. Note that, the larger average KL-divergence, the higher quality of the clusters.

Single-Type vs. Multi-Type We compare the quality of clusters detected using multi-typed attribute objects with those detected using single-typed attribute objects in static networks. For the segment G_5 (i.e., year 2006 to 2009), we select different combinations of attribute types for learning the clusters and compare the results of the generated clusters, which is summarized in Table 2. The attribute types used for modeling clusters are listed in the first column, the compactness score for target objects is computed accordingly. We then calculate the average KL-divergence for each type of attributes and their overall mean. For the cases using only single type or partial types of attribute objects, the distribution parameters for all other types can also be obtained according to the target objects in each cluster. For example, in the first row of Table 2, we model and learn the clusters only through the authors linked to papers, but once we get the papers in each cluster, we can get the count of conferences and terms accordingly for each cluster. We do not put single venue in the table, since one cannot cluster papers merely according to its published venue as one paper goes to one venue exactly.

TABLE 2
Cluster Qualities under Different Attribute Type Combinations

Attr. Type	$Compa.$	$avgKL_C$	$avgKL_A$	$avgKL_W$	$avgKL_{all}$
A	1.0153	0.0735	3.3411	0.7697	1.3948
W	0.9952	0.5836	1.6649	**2.5658**	1.6048
C+A	1.2135	2.6390	**3.4382**	1.1055	2.3942
C+W	2.0016	2.6819	2.3187	2.5264	2.5090
A+W	1.7606	1.2398	2.7205	2.4060	2.1221
C+A+W	**3.1158**	**3.3135**	3.0007	2.3521	**2.8888**

TABLE 3
Cluster Qualities under Different Historical Impacts

Net. Seq.	$Compa.$	$avgKL_C$	$avgKL_A$	$avgKL_W$	$avgKL_{all}$
G_5	3.1158	3.3135	**3.0007**	**2.3521**	**2.8888**
G_4-G_5	3.9076	**3.6770**	2.4924	2.3772	2.8489
G_3-G_5	3.6753	3.1765	2.6869	2.3070	2.7224
G_2-G_5	4.1364	3.2095	2.6424	2.2723	2.7081
G_1-G_5	**4.3120**	3.2953	2.3818	2.2729	2.6500

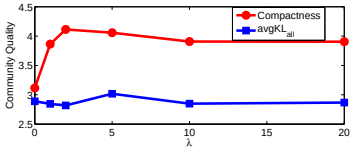

Fig. 5. The Impacts of Smoothing Parameter λ

Table 2 shows that the more types of the attribute objects we use, the better quality the detected clusters, in both NMI and $avgKL_{all}$ measures. Notice that by merely using some single type of attribute objects, say terms (W), the value for this type, $avgKL_W$, could be high. However, this causes ***overfitting for the particular type***, and therefore reduces the overall performance. That is the reason why we use multiple types of objects to model the target objects collectively.

Static vs. Evolution We compare the quality of clusters detected in static networks with those detected in an evolution setting (Table 3). We study how prior clusters in previous time windows can improve the quality of the clusters detected in the current time. Still considering the cluster quality detected in G_5, we vary the length of history information in detecting these clusters. In other words, we run *Evo-NetClus* algorithm for network sequences of different lengths that end at G_5, and compare the cluster quality between them. The results show that using longer history information in general produces higher cluster quality for G_5 in terms of compactness scores for target objects. However, the overall $avgKL_{all}$ has an inverse trend. This implies that detecting clusters by merely using networks observed in current time window may cause ***overfitting for the particular time period***, and affect the model performance.

Further, we study the impact of smoothing hyper-parameter λ to the cluster qualities. Recall that the smoothing rate given λ and γ is $\frac{\lambda}{\gamma+\lambda}$. The results are shown in Fig. 5, from which we can see that λ should be set as a trade-off between the current network and the previous one, though different evaluation measures indicate different preferences.

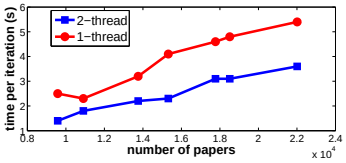

Fig. 6. Efficiency Test of Evo-NetClus Algorithm

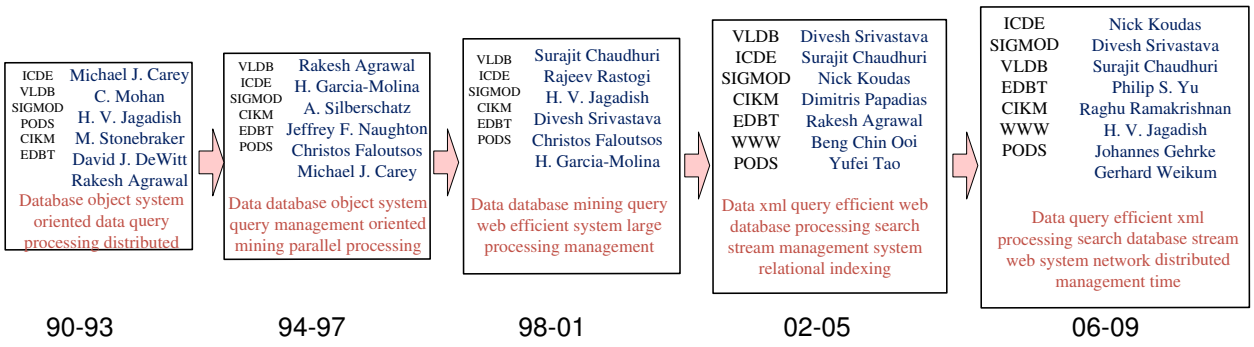

Fig. 7. The Co-Evolution of Conference, Authors and Terms within Database Cluster

5.3 Efficiency Study

We test the efficiency of *Evo-NetClus* on the DBLP network sequences with the time window length as 1 from year 1991 to 1997, each with the former year as the prior year. The running time (seconds) per iteration versus the number of papers in networks at each year is shown in Fig. 6. We also implement a multi-threading version for the algorithm. Our algorithm is nearly linear to the number of target objects in the network, and the 2-thread version of the algorithm can achieve an average speed up of 1.55 times. In most cases, *Evo-NetClus* converges within 50 iterations.

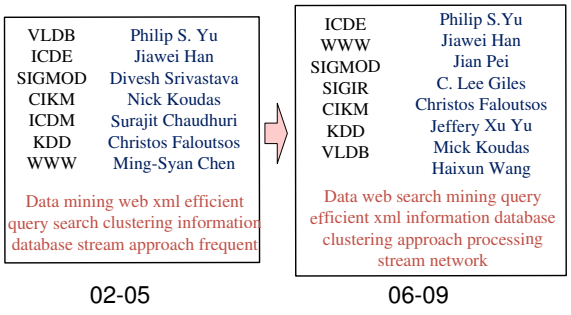

Fig. 8. Evolution of DB Cluster by Using Only Terms

5.4 Case Studies

We use case studies to show the co-evolution of multiple types of objects within clusters and the evolutionary structure between the clusters.

DBLP Network Case Study Fig. 1 already gave a partial evolution structure over the whole computer science area in DBLP from 1990 to 2009, with the length of time window set to 2. Now, we use the "four-areas" dataset to show the evolution in a finer scale. As it is difficult to show the whole T-partite graph (here $T = 5$), we output the partial structure in a query mode.

First, we output the co-evolution of authors, conference, and terms within the cluster "database" (the label "database" is added to the cluster only afterwards, Fig. 7), and compare it with the results merely using terms in the modeling (Fig. 8). We can see that using only terms, one cannot detect pure database clusters that are as good as using all the attribute types.

Second, we trace back the cluster of "web search and web service" ("web" in short) and plot the evolutionary structure between it and other related clusters in a tree structure (Fig. 9). In Fig. 9, each square represents a cluster, where the labels for the clusters are shown with their cluster size (the number of target objects in the cluster). The links are displayed between other clusters and the "web" cluster, if the cluster is among the top-3 contributors for the "web" cluster. The weight of the link between clusters A and B means the number of papers belonging to A as the prior group but are assigned to cluster B after learning. One can see that the "web" cluster emerges in the time window $t = 3$ (year 1998 to 2001), as no prior clusters are major contributors to it. It then absorbs contents from the web cluster in database conferences and database cluster and grows in time window 4. After that, it further absorbs contents from database and data mining cluster, and grows into an even bigger cluster.

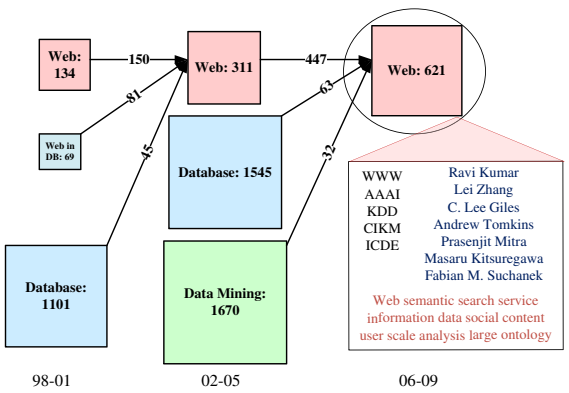

Fig. 9. Evolutionary Tracing Tree for Web Cluster

Twitter Network Case Study For the 8-week Twitter network sequence, by setting $\beta_{hashtag} = \beta_{term} = 0.1$ and $\beta_{user} = 0.05$, we obtain around 10-12 clusters each week. We follow two clusters: "news" and "soccer", and display their top attribute objects in Table 4 (a) and (b). The "news" cluster is a relatively stable cluster in terms of its size and participants from different types of objects. We can correctly figure out the hot trends each week, such as Japan nuclear crisis in middle March, royal wedding in late

TABLE 4
Evolution of Clusters in Twitter Network Sequence

(a) "News" Cluster

Week	3/13 - 3/19	3/20 - 3/26	...	4/24 - 4/30	5/1 - 5/7
Cluster size	15728	19299	...	24189	24472
Top Hashtags	job japan libya tcot bahrain nuclear	job libya tcot japan hiring egypt	...	royalwedding job nfldraft tcot elxn41 syria	job elxn41 tcot obl osama travel
Top Terms	japan news online march business nuclear social	news online march business japan blog libya	...	news online wedding business post royal money	news online business post obama bin laden
Top Users	mashable reuters breakingnews youtube guardian huffingtonpost	mashable digg breakingnews wnblibya reuters addtoany	...	youtube icreate_id mashable technobuffalo pddekock digg	youtube mashable gmail huttingtonpost technobuffalo guardian

(b) "Soccer" Cluster

Week	...	4/3 - 4/9	...	4/24 - 4/30	5/1 - 5/7
Cluster size	N/A	1209	...	1747	1670
Top Hashtags	N/A	youright wizard rionocineartcafedia8 mufc jackmarshallrules motogp	...	youright mufc letsgopens tragicmoviedeaths football cfc	mufc football manutd cfc ynwa championsleague
Top Terms	N/A	rafael chelsea inter united schalke tan milan	...	madrid barca barcelona united messi schalke arsenal	madrid united barca barcelona final milan manchester
Top Users	N/A	duniasoccer vinyciusn demilmylife soniaabrao restertempe mayara_no	...	tweetramalan detiksport detikcom duniasoccer stafanyls allsmiler	tweetramalan detikcom duniasoccer detiksport onlyls metro_tv

April, and Bin Laden got killed in early May. Notice that, "Bin Laden" topped in many other clusters that week. Our algorithm chooses not to merge these clusters although they talk about some common topic for a while. By studying co-evolution of different types of objects, our algorithm can correctly detect the hidden clusters even if they have overlaps in some types of objects. The "soccer" cluster does not exist in the first 3 weeks, and it appears in the week of April 3 - April 9, when the first quarter final of 2010-11 UEFA Champions League starts. By looking more into the top objects, we can see the team names, player names, coach names, and so on.

In Table 5, we list 3 clusters detected under *Evo-NetClus* and *Evo-Text* respectively for the Twitter network for Week 8. For each cluster, we only show the top-10 terms. By using *Evo-NetClus*, we can discover consistent clusters such as "news", "digital products", "food" and so on. However, the clusters detected by *Evo-Text*, which only uses terms for clustering, are generally with poorer quality. For example, "news" and "digital products" are mixed together in Cluster 5, due to that *Evo-Text* has not further used other types of objects (such as users) to distinguish the two clusters and text information in each tweet is quite short. "Food" cluster has not been found in *Evo-Text*, instead, a "weather" cluster is detected due to the overfitting by using only terms. However, in reality weather is usually mentioned as background in different clusters and should not be treated as a cluster by itself. For clusters that are with much more focused terms, such as "soccer" cluster, *Evo-Text* and *Evo-NetClus* output similar top-k terms.

Delicious Network Case Study For the half-year Delicious dataset, we track a cluster related to "wikileaks", and show the co-evolution of websites and tags in the cluster. Users are not shown here as we can hardly read information from the user IDs. As the urls are usually too long for display, we only show the last segment of the urls. We can see that as the cablegate event happened on Nov. 28, a cluster was formed to discuss the related issues on that week; then the arrest of Assange in the next week made

TABLE 5
Comparison of Clusters in Twitter Network $T = 8$
(Only Showing Top-10 Terms)

(a) Evo-NetClus

ID:5	ID:10	ID:2
news	battery	ang
online	iphone	ice
wedding	bir	cream
business	frame	eat
post	laptop	chicken
royal	black	chocolate
money	apple	eating
blog	ipad	easter
april	wind	dinner
guide	case	cheese

(b) Evo-Text

ID:4	ID:6	ID:9
news	ang	madrid
online	wind	barcelona
business	humidity	united
iphone	mph	barca
money	ako	schalke
battery	rain	arsenal
post	harry	league
blog	draft	messi
guide	temperature	final
apple	potter	goal

the size of the cluster into peak; after that the cluster kept for another 2 weeks and evolved to a cluster discussing reviews of 2010 in the end of the year. Besides the event-based clusters, we also have detected relative stable clusters like "electronic devices" and "web design." Compared to the DBLP network, clusters in Delicious are overall more dynamic and varied.

6 Related Work

The problem of community detection and evolution has been studied extensively with numerous methods proposed, such as modularity-based community discovery [17], [16], clustering-based community detection [27], [28], probabilistic model-based network analysis [20], [9], [7], [1], and dynamic community analysis [5], [11], [3], [19], [6], [8]. However, most of the existing methods are on *homogeneous* networks, with very few on detecting evolutionary clusters and co-evolving different types of objects in dynamic *heterogeneous* information networks.

Two recent studies [24], [12] are on cluster detection and evolution in heterogeneous networks. However, their approaches are matrix factorization-based methods, and encounter difficulties at handling real-world networks. First, it is difficult for them to handle dynamic variation of object sets. Second, they need users to specify the number

TABLE 6
Evolution of the "Wikileaks" Cluster in Delicious Network Sequence

Time Win. (Size)	Top Urls	Top Tags
11/25-12/01 (1065)	4thamendment; tsa-x-ray-backscatter-body-scanner.html; us-embassy-cables-wikileaks; us-embassy-cable-leak-diplomacy-crisis; julian-assange-and-the-computer-conspiracy	wikileaks politics tsa science security internet health government travel journalism cablegate privacy economics
12/02-12/08 (1258)	dont-shoot-messenger-for-revealing-uncomfortable-truths;julian-assange-wikileaks; western-democracies-must-live-with-leaks; after_secrets;wikileaks.ch/mirrors.html	wikileaks politics journalism internet government cablegate media news security assange democracy censorship
12/09-12/15 (1028)	more_wikileaks; wikileaks-cables-shell-nigeria-spying; climate-change.html; wikirebels_the_documentary; wikileaks-backlash-cyber-war	wikileaks politics internet travel history video privacy documentary security culture media government
12/16-12/22 (803)	anonymous-wikileaks-protest-amazon-mastercard; julian-assange-sweden;save_tweets;the-political-power-of-social-media	wikileaks politics phd 2010 journalism internet news education photography economics guardian data media
12/23-12/29 (594)	demolition_of_the_paris_metro; 2010-year-review; the-political-power-of-social-media; year-review-2010.cfm; 2010_in_photos_part_1_of_3.html	politics culture 2010 video photography internet education history economics science media news research wikileaks

of clusters for each time window and cannot model the evolutionary structure among clusters. In this study, we adopt statistical methods that can add prior knowledge for different types of objects and automatically determine the best clusters without specifying the numbers of them.

Dirichlet process [15], [26] provides a way to add prior on the parameters of a mixture model, and is helpful to decide the number of clusters automatically. Recently, some studies, such as in [33] and [18], have extended the Dirichlet process to consider time information. Some other DP-based extensions [29], [30], [2] have been proposed to model evolutionary clustering. However, these methods are designed for traditional (i.e., numerical value or text) attribute-based clusters, and cannot be directly applied to our problem setting, where a cluster is a multi-typed cluster in a heterogeneous network. Also, many of these studies [2] model only limited evolution structures, such as birth, death and evolving, while neglecting other evolution structures such as split and merge. Dirichlet Process has recently been used to model blockmodel-based link generation, where the clusters of nodes can be infinite and dynamically evolved [10]. However, it can only output evolutional communities for homogeneous networks, and it is unclear how this model can be extended to model co-evolution yet.

Another line of work related to evolutionary network clustering is evolutionary topic modeling, which tries to extract the best topic models at each timestamp that satisfy the constraints of temporal smoothness, such as in [4], [13], [32]. However, merely studying the evolution of topics without considering the link information in the networks cannot fully reflect the evolution of clusters: A cluster containing different types of objects carries more semantic information, and links in the network can tell more about the connections between clusters and can help detect more accurate evolutionary clusters. In this paper, we study the co-evolution of multi-typed objects.

Our recent work [22] studied the community evolution in heterogeneous networks. However, it can only model limited evolutionary structure such as birth, death, evolving, and split, while in real-world network cluster evolution also involves structures such as merge. In this paper, we extend the previous model such that it can model more complex dependency structure between clusters, and can output a full T-partite graph as the evolutionary structure.

7 Conclusions

In this paper, we have studied the problem of co-evolution of multi-typed objects in dynamic heterogeneous information networks with star network schemas. A Hierarchical Dirichlet Process Mixture Model-based generative model *Evo-NetClus* is proposed to model the cluster generation process given the prior clusters in the previous time window. An efficient greedy algorithm is proposed to learn the model. Experiments on the DBLP, Twitter, and Delicious datasets have demonstrated the effectiveness and efficiency of the algorithm. It shows shows that modeling the co-evolution of objects from multiple types indeed enhances the cluster quality. The case study shows that the evolutionary structure so detected are meaningful and interesting.

Acknowledgment

The work was supported in part by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), NSF IIS-0905215, CNS-09-31975, and CCF-0905014 and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC. Jie Tang is supported by the Natural Science Foundation of China (No. 61222212, No. 61073073), and a research fund from Huawei Inc.

References

[1] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing. Mixed membership stochastic blockmodels. *J. Mach. Learn. Res.*, 9:1981–2014, 2008.
[2] A. Ahmed and E. P. Xing. Dynamic Non-Parametric Mixture Models and the Recurrent Chinese Restaurant Process: with Applications to Evolutionary Clustering. *SDM'08*, 219–230, 2008
[3] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *ACM Trans. Knowl. Discov. Data*, 3:16:1–16:36, 2009.
[4] D. M. Blei and J. D. Lafferty. Dynamic topic models. *ICML'06*.
[5] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. *KDD'07*.
[6] W. Fu, L. Song, and E. P. Xing. Dynamic mixed membership blockmodel for evolving networks. *ICML'09*.
[7] M. S. Handcock, A. E. Raftery, and J. M. Tantrum. Model-based clustering for social networks. *J. Royal Stat. Soc. A*, 170(2):301–354, 2007.
[8] Q. He, B. Chen, J. Pei, B. Qiu, P. Mitra, and L. Giles. Detecting topic evolution in scientific literature: how can citations help? *CIKM'09*.

[9] P. D. Hoff, A. E. Raftery, and M. S. Handcock. Latent space approaches to social network analysis. *J. Am. Stat. Assoc.*, 97:1090–1098, 2001.
[10] K. Ishiguro, T. Iwata, N. Ueda, and J.B. Tenenbaum. Dynamic Infinite Relational Model for Time-varying Relational Data Analysis. *NIPS'10*, 919-927.
[11] M.-S. Kim and J. Han. A particle-and-density based evolutionary clustering method for dynamic networks. *VLDB'09*.
[12] Y.-R. Lin, J. Sun, P. Castro, R. Konuru, H. Sundaram, and A. Kelliher. Metafac: community discovery via relational hypergraph factorization. *KDD'09*.
[13] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. *KDD'05*.
[14] R. M. Nallapati, A. Ahmed, E. P. Xing, and W. W. Cohen. Joint latent topic models for text and citations. *KDD'08*.
[15] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *J. Comput. & Grap. Stat.*, 9(2):249–265, 2000.
[16] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Phys. Rev. E*, 74:036104, 2006.
[17] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2), 2004.
[18] L. Ren, D. B. Dunson, and L. Carin. The dynamic hierarchical dirichlet process. *ICML'08*.
[19] P. Sarkar and A. W. Moore. Dynamic social network analysis using latent space models. *SIGKDD Explor.*, 7(2):31–40, 2005.
[20] T. A. B. Snijders. Markov chain monte carlo estimation of exponential random graph models. *J. Social Structure*, 3, 2002.
[21] A. Strehl and J. Ghosh. Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.*, 3:583–617, 2003.
[22] Y. Sun, J. Tang, J. Han, M. Gupta, and B. Zhao. Community evolution detection in dynamic heterogeneous information networks. *MLG'10*.
[23] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. *KDD'09*.
[24] L. Tang, H. Liu, J. Zhang, and Z. Nazeri. Community evolution in dynamic multi-mode networks. *KDD'08*.
[25] Y. W. Teh. Dirichlet processes. In *Ency. Machine Learning*. Springer, 2010.
[26] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Hierarchical Dirichlet processes. *J. Amer. Stat. Assoc.*, 101(476):1566–1581, 2006.
[27] U. von Luxburg. A tutorial on spectral clustering. Tech. report, Max Planck Inst. for Biological Cybernetics, 2006.
[28] S. White and P. Smyth. A spectral clustering approach to finding communities in graph. *SDM'05*.
[29] T. Xu, Z. M. Zhang, P. S. Yu, and B. Long. Dirichlet process based evolutionary clustering. *ICDM'08*.
[30] T. Xu, Z. M. Zhang, P. S. Yu, and B. Long. Evolutionary clustering by hierarchical dirichlet process with hidden markov state. *ICDM'08*.
[31] C. Zhai and J. D. Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214, 2004.
[32] J. Zhang, Y. Song, C. Zhang, and S. Liu. Evolutionary hierarchical dirichlet processes for multiple correlated time-varying corpora. *KDD'10*.
[33] X. Zhu, Z. Ghahramani, and J. Lafferty. Time-sensitive dirichlet process mixture models. Tech. report, 2005.

Yizhou Sun received her Ph.D. from UIUC in 2012. She is an assistant professor in the College of Computer and Information Science of Northeastern University. Her principal research interest is in mining information and social networks, and more generally in data mining, database systems, statistics, machine learning, information retrieval, and network science. Yizhou has over 40 publications in books, journals, and major conferences. Tutorials based on her thesis work on mining heterogeneous information networks have been given in several premier conferences, including EDBT'09, SIGMOD'10, KDD'10, ICDE'12, VLDB'12, and ASONAM'12. She received ACM KDD 2012 Best Student Paper Award.

Jie Tang is an associate professor at the Department of Computer Science and Technology, Tsinghua University. His main research interests include data mining algorithms and social network theories. He has been visiting scholar at Cornell University, Chinese University of Hong Kong, Hong Kong University of Science and Technology, and Leuven University. He has published over 100 research papers in major international journals and conferences including: KDD, IJCAI, WWW, SIGMOD, ACL, Machine Learning Journal, TKDD, TKDE, and JWS.

Jiawei Han Bliss Professor of Computer Science, University of Illinois at Urbana-Champaign. He has been researching into data mining, information network analysis, database systems, and data warehousing, with over 600 journal and conference publications. He has chaired or served on many program committees of international conferences, including PC co-chair for KDD, SDM, and ICDM conferences, and Americas Coordinator for VLDB conferences. He also served as the founding Editor-In-Chief of ACM Transactions on Knowledge Discovery from Data and is serving as the Director of Information Network Academic Research Center supported by U.S. Army Research Lab. He is a Fellow of ACM and IEEE, and received 2004 ACM SIGKDD Innovations Award, 2005 IEEE Computer Society Technical Achievement Award, 2009 IEEE Computer Society Wallace McDowell Award, and 2011 Daniel C. Drucker Eminent Faculty Award at UIUC. His book "Data Mining: Concepts and Techniques" has been used popularly as a textbook worldwide.

Cheng Chen is a first year graduate student in CSAIL at MIT. He works with Prof. Shafi Goldwasser in Theory group. Before this, Cheng got his B.S degree in Computer Science from Yao Class, Institute for Interdisciplinary Information Sciences, Tsinghua University in 2012.

Manish Gupta is an applied researcher at Microsoft Bing, India. He received his Masters in Computer Science from IIT Bombay in 2007 and his PhD in Computer Science from Univ of Illinois at Urbana Champaign in 2013. He worked for Yahoo! Bangalore from 2007 to 2009. His research interests are in the areas of data mining and information retrieval.