RC25012 (W1006-064) June 17, 2010
Computer Science

IBM Research Report

Evolutionary Clustering and Analysis of Heterogeneous Information Networks

Manish Gupta[1], Charu Aggarwal[2], Jiawei Han[1], Yizhou Sun[1]

[1]University of Illinois at Urbana Champaign

[2]IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Cambridge - Haifa - India - T. J. Watson - Tokyo - Zurich

Evolutionary Clustering and Analysis of Heterogeneous Information Networks

Manish Gupta
University of Illinois at Urbana Champaign
manishg.iitb@gmail.com

Charu Aggarwal
IBM T. J. Watson Research Center, NY
charu@us.ibm.com

Jiawei Han
University of Illinois at Urbana Champaign
hanj@cs.uiuc.edu

Yizhou Sun
University of Illinois at Urbana Champaign
sun22@illinois.edu

ABSTRACT

In this paper, we study the problem of evolutionary clustering of multi-typed objects in a heterogeneous bibliographic network. The traditional methods of homogeneous clustering methods do not result in a good typed-clustering. The design of heterogeneous methods for clustering can help us better understand the evolution of each of the types apart from the evolution of the network as a whole. In fact, the problem of clustering and evolution diagnosis are closely related because of the ability of the clustering process to summarize the network and provide insights into the changes in the objects over time. We present such a tightly integrated method for clustering and evolution diagnosis of *heterogeneous bibliographic* information networks. We present an algorithm, ENetClus, which performs such an agglomerative evolutionary clustering which is able to show variations in the clusters over time with a temporal smoothness approach. Previous work on clustering networks is either based on homogeneous graphs with evolution, or it does not account for evolution in the process of clustering heterogeneous networks. This paper provides the first framework for evolution-sensitive clustering and diagnosis of heterogeneous information networks. The ENetClus algorithm generates consistent typed-clusterings across time, which can be used for further evolution diagnosis and insights. The framework of the algorithm is specifically designed in order to facilitate insights about the evolution process. We use this technique in order to provide novel insights about bibliographic information networks.

1. INTRODUCTION

Information networks have become ubiquitous in recent years because of the large number of networked applications such as social networks, the web and other linked entities. For example, academic networks such as DBLP, biological networks, and massive entity-relation models are all examples of information networks. Such networks have the common property that they contain different kinds of entities which interact with one another. Some of the examples such as social networks and the web are inherently homogeneous, since they contain entities of the same type. For example, a social network contains actors that are linked by friendship relationships, whereas the web contains documents which are linked by hyper-links.

Most work in the area of graph and network mining has focused on the homogeneous domain. However, a heterogeneous representation is much richer; for example a richer representation of a bibliographic network may contain nodes corresponding to different entities such as author, conference, paper and term. Edges may denote more diverse relationships such as written-by between an author node and a paper node, contains between a term node and a paper node, published-in between a paper node and a conference node. Clearly, the richer representation of a heterogeneous network makes it powerful; on the other hand it is also much more challenging for mining purposes. In recent years, there has been an increasing interest in the area of heterogeneous information networks. In this paper, we will examine the problem of clustering and evolution diagnosis in massive information networks.

Heterogeneous information networks are often encountered in dynamic environments which are continuously evolving. The problem of clustering has been studied recently in the context of non-evolving and static information networks. E.g., Sun et al. [15] present NetClus, which is a clustering algorithm for *star schema-based heterogeneous information networks*. The center of the star is termed as the *target type* while other type nodes are connected to this center type and are called as *attribute type* nodes. E.g. for the DBLP graph, paper is a target type while authors, conference, terms are attribute type nodes. Figure 1 illustrates a net-cluster view of the DBLP network. The star-schema is a particularly important case, because of its representational power in a variety of scenarios.

While NetClus is a powerful algorithm for determining heterogeneous clusters, it incorporates no notion of evolution. NetClus, if used over multiple snapshots, would produce clusters which would have no correspondence with the ones in the previous snapshot. This effect becomes much more prominent because NetClus clusters depend immensely on the initial seeds. The problem of evolutionary clustering has been studied in the context of *homogeneous networks* [4]. The basic principle is to create a clustering which focuses on both maintaining high quality clusters, and on creating clusters in which a natural correspondence can be maintained among the clusters across different snapshots.

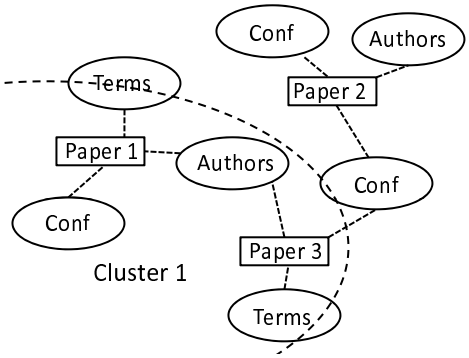

Figure 1: Net-cluster view of DBLP network

This broad technique is referred to as *temporal smoothing*. A variety of techniques [5, 12] have been designed for determining evolutionary clusters using temporal smoothing. However, these techniques are inherently designed for the homogeneous case. For heterogeneous networks, we wish to cluster entire entity (group of related objects) as a whole rather than clustering of individual types separately. It is challenging to cluster the entities (each consisting of multiple types of nodes) with temporal smoothness such that the snapshot quality is maintained.

In this paper, we will study the problem of evolutionary clustering and diagnosis in information networks. We will take a broader view of clustering and evolution analysis as two tightly integrated problems which can be used in order to derive interesting insights from data. This is especially true in the case of heterogeneous information networks, since one can study how the trends in the different kinds of entities are affected by one another. For example, in an authorship network, evolutionary clustering can be combined with careful evolutionary diagnosis and metrics to determine merges and splits of different topical areas, authorship evolution and topical evolution. Such insights are critical in understanding the nature of the changes which occur in dynamic and rapidly evolving information networks. Since clustering can be viewed as a network summarization technique, it is a natural approach for integrating with the problem of evolutionary diagnosis in order to understand and summarize the key changes which may occur in a network over time.

In this paper, **we make the following contributions**:

1. While the problem of evolutionary clustering has been studied for the homogeneous case, the problem is much more challenging for the heterogeneous case, because of the different entity-types which may evolve over time. This paper is the first to present an evolutionary clustering algorithm for heterogeneous networks. Our algorithm returns temporally smoothed, high quality agglomerative clusters, and leverages on some concepts derived from the NetClus framework.

2. We tightly integrate the problems of evolutionary clustering and diagnosis; the evolutionary diagnosis is achieved by defining metrics and techniques to characterize the clustering behavior over time. For example, we can design techniques to identify the birth, continuation and slow disappearance of a community. We also study the influence of one community onto another and flux between two different kinds of communities. Our techniques are general enough to deal with different time granularities and entity types.

3. One of the additional results of this effort is to provide novel insights into the evolution of bibliographic networks with the use of the techniques proposed in this paper. As a specific example, we use the DBLP dataset in order to provide novel insights about the evolution process.

The paper is organized as follows. In Section 2, we first present our extension to the NetClus framework. In Section 3, we define evolution diagnosis methods and metrics. We present experimental results on the DBLP dataset in Section 4. We then present an overview of related work in Section 5. We conclude with a summary and future work in Section 6.

2. THE ENETCLUS ALGORITHM

In this section, we present our algorithm for clustering of an evolutionary information network. The broad approach is to use a probabilistic generative model in which we model the probability of generation of different objects from each cluster. A maximum likelihood technique is used to evaluate the posterior probability of presence of an object in a cluster. The conditionals (i.e., probability of the presence of an object in a cluster) are computed using ranking of object within current clusters and representativeness from previous clustering. The priors (probability of clusters) are estimated using an Expectation Maximization approach. We first describe the problem formulation and the methods for computing the underlying probabilities. Then we describe the ranking and the clustering parts of the algorithm, with a special emphasis on how the dynamic evolution affects different parts of the NetClus framework.

2.1 Problem Formulation

Given different snapshots of a graph, each of which contain nodes of multiple types, our aim is to find a consistent agglomerative clustering of the graph snapshots across time. Consistency refers to our ability to relate the clusters to one another in different snapshots, so that it may be better possible to diagnose the evolution process. Let GS denote the graph sequence $\{G_i\}_{i=1}^{N}$ where each of the graphs G_i is a snapshot taken at the time instant $\{y_i\}_{i=1}^{N}$. Given the number of levels L and the number of clusters K, we would like to obtain a net-cluster tree sequence CTS for the graph sequence GS.

Definition 2.1 (Net-Cluster). *A net-cluster c of a graph $G(V, E)$ is a subgraph $G'(V', E')$ such that $V' \subseteq V$ and $E' \subseteq E$ and $\forall e \in E$, weight $W_e(E) = W_e(E')$. Let $b_c : V' \rightarrow [0, 1]$ denote the probability with which an object $o \in V'$ belongs to cluster c. If o is a target type object, $b_c(o)$ is either 0 or 1; for attribute type objects o, $b_c(o) \in [0, 1]$.*

Definition 2.2 (Net-Cluster Tree). *A net-cluster tree CT for a graph G is a tree with L levels (level 1 being the root and level L being the leaves) and branching factor K. Root of CT corresponds to the graph G. Children nodes $\{nc_i\}_{i=1}^{K}$ of a node n store the K net-clusters $\{c_i\}_{i=1}^{K}$ obtained as a result of the clustering of the subgraph $G_n(V_n, E_n)$ at node n. The distribution of the target objects within each of the children nodes of a node n forms a disjoint partition of target objects in node n. An attribute type object $o \in V_n$ belongs to the child node nc_i with probability $b_i(o)$. Children nodes of a node n are ordered.*

Definition 2.3 (Net-cluster Tree Sequence). *A net-cluster tree sequence CTS corresponding to a graph sequence*

GS is a sequence of net-cluster trees $\{CT_i\}_{i=1}^{N}$ *where* CT_i *corresponds to time instant* y_i*. Each of the trees in the sequence have the same branching factor and same number of levels. The children of every node in each of the trees is ordered in the sense that first child of a node* n_i *in* CT_i *corresponds to the first child of a similar node* n_j *in* CT_j*.*

We aim at generating such a net-cluster tree sequence CTS for a graph sequence GS such that the trees are consistent and represent high-quality clusters.

2.2 ENetClus Framework

To perform evolutionary clustering in a heterogeneous network, one could use any of the homogeneous clustering algorithms to cluster each type of nodes individually. But that would not guarantee that all the objects related to same entity lie in the same cluster and also the mutual information between different types of objects would not be exploited. To achieve agglomerative temporally smoothed clusters, we exploit a natural variation of the NetClus algorithm. NetClus performs iterative ranking and clustering. We use the knowledge from the current snapshot clustering to initialize the clusters for the next snapshot and also to influence the ranking of objects in these new clusters using the priors. Note that the priors are propagated in the forward direction as time progresses. This means that the algorithm can be executed as an online evolutionary clustering algorithm. Supplying good priors actually improves the quality of NetClus algorithm. Thus, with greater consistency, we achieve better quality unlike other evolutionary clustering algorithms. Algorithm 1 shows our framework. We explain each of the steps in further subsections.

Algorithm 1 NetClus with Evolution-Aware Priors

1: Priors: Initialize prior probabilities $\{P(o|c_k)\}_{k=1}^{K}$.
2: Initialize: Generate initial net-clusters. $\{c_k^0\}_{k=1}^{K}$.
3: Rank: Build probabilistic generative model for each net-cluster, i.e., $\{P(o|c_k^t)\}_{k=1}^{K}$.
4: Cluster-target: Compute $(p(c_k^t|o))$ for target objects and adjust their cluster assignments.
5: Iterate: Repeat steps 3 and 4 until the clusters don't change significantly.
6: Cluster-attribute: Calculate $p(c_k^*|o)$ for each attribute object in each net-cluster.
7: **return** $p(c_k^*|o)$

2.3 Initialization of Priors and Net-Clusters

At the first snapshot, which is denoted by y_1, prior probabilities are defined intuitively. E.g., if we believe that the data has 4 clusters, we can define high prior probabilities for the terms representative of each cluster. For other time instants, the prior probabilities $\{P(o|c_k)\}_{k=1}^{K}$ are defined as follows. The prior probability of an object o belonging to cluster c_k is defined as its representativeness in the corresponding cluster within the net-cluster tree for the previous time instant (step 1). The use of these priors ensures temporal smoothness, because the computation of cluster membership in a particular snapshot is affected by the membership behavior in previous snapshots. The representativeness of an object o in cluster c depends on the probability of generating that object (given cluster c), and is inversely proportional to the entropy of the cluster membership distribution of the object o. An object with a distribution peaked at cluster c will have a high representativeness value for cluster c. Subsection 2.4 illustrates how the ranking part of the algorithm uses these priors.

The initialization of clusters should be done in a way that provides a smooth transition from the clustering in the previous snapshot. Hence, the algorithm generates initial partitions for target objects as follows. Let $\{o_i\}_{i=1}^{L}$ be the L attribute type objects connected to an object o. Consider a probability distribution using the priors mentioned in step 1, $\{p_k = \sum_{i=1}^{L} P(o_i|c_k)\}_{k=1}^{K}$. A target object o is assigned to cluster c_k with max probability p_k. Then initial net-clusters are induced from the original network according to these partitions, i.e., $\{c_k^0\}_{k=1}^{K}$. This corresponds to step 2 of the algorithm. Initializing clusters using representativeness values ensures faster convergence to a better local maxima.

2.4 Ranking and Clustering

In step 3, the algorithm builds ranking-based probabilistic generative model for each net-cluster, i.e., $\{P(o|c_k^t)\}_{k=1}^{K}$. The ranking process constructs representative objects from the different clusters. The corresponding probability can be decomposed by conditioning on the type of the object in the corresponding cluster. In other words, we have $P(o|c_k) = P(T_o|c_k) \times P(o|T_o, c_k)$. $P(T_o|c_k)$ is estimated as the maximum likelihood estimate of type T_o in cluster c_k. $P(o|T_o, c_k)$ can be computed using two different notions of ranking. For some attributes, we can use a frequency-based approach of estimating this probability. For example, in a bibliographic information network, the probability for a term is the weighted ratio of number of papers containing this term to the total number of papers, where weights are associated with every paper. $P(o|T_o, c_k)$ can also be computed using authority-based ranking. In this case, ranking of an object o is estimated by propagating authority scores from objects of other attribute types via the target type. Finally, the overall probability $P(o|T_o, c_k)$ is computed as a weighted sum of ranking-based $P(o|T_o, c_k)$ and the priors generated in step 1. The prior weight (λ_P) controls how much the current ranking and therefore the current clustering depends on the clustering at the previous time instant. Thus, λ_P controls the tradeoff between the snapshot clustering quality and temporal smoothness.

By using the ranked attribute object probabilities, we can compute the conditional probability of a target object o as $P(o|c_k) = \prod_{x \in N_{G_k}(o)} P(T_x|c_k)^{W(o,x)} \times P(x|T_x, c_k)^{W(o,x)}$, where N_{G_k} is the neighborhood set of objects in subgraph G_k.

In step 4, the algorithm calculates the posterior probabilities for each target object. This is done by iterating over the EM equations $p^t(c_k|o) \propto p(o|c_k) \times p^t(c_k)$ and $p^{t+1}(c_k) = \sum_{i=1}^{|O|} p^t(c_k|o_i)/|O|$.

Once the posterior probabilities $P(c_k|o)$ have been computed, these can be used to express the object o as a vector $v_o = (p(c_1|o), p(c_2|o), ...p(c_K|o))$. This new vector space can be leveraged for similarity computation and object assignment. By using previous cluster assignments, vector v for the cluster centroids is computed as an average of objects belonging to that cluster. The object o is re-assigned to a cluster, by using the cosine similarity value between v_o and cluster centroids.

The steps 3 and 4 are repeated until the clusters do not change significantly, i.e., $\{c_k^*\}_{k=1}^{K} = \{c_k^t\}_{k=1}^{K} \approx \{c_k^{t-1}\}_{k=1}^{K}$ in

the K-dimensional vector space.

Finally, in step 6, posterior probabilities for each attribute object ($p(c_k^*|o)$) in each net-cluster are computed using the posterior probabilities of its neighboring target objects. Note that the way the priors are propagated automatically ensures matching of the clusters at all levels in the hierarchy. This is another advantage of our maximum likelihood based model compared to other evolutionary clustering schemes where greedy methods are used to find corresponding clusters across different snapshots.

2.5 Complexity Analysis

As mentioned in [15], clustering once requires $O(c_1|E| + c_2N)$ time, where N is the number of target objects. A net-cluster tree with L levels is created after clustering of all internal nodes i.e., $\frac{K^{L-1}-1}{K-1}$ nodes. But, on an average, the size of the graph at a node decreases K times per level. Hence, the creation of a cluster tree requires $O(L \times (c_1|E| + c_2N))$ time. The time required to compute the entire CTS would also depend on the number of time granularities and the number of time instants per time granularity. The exact complexity would depend on how dense the graph becomes at different instants and intervals of time.

3. EVOLUTION DIAGNOSIS AND METRICS

In this section, we discuss methods for evolution diagnosis and metrics. We define metrics for evolution quantification such as appearance and disappearance rate, continue/merge/split rate, stability and sociability of objects. We further note that an algorithmic tradeoff exists between clustering quality and the consistency of the clusters over time. While most of the properties studied in this section are properties of the *data*, the latter are the properties of the algorithm in terms of the level of smoothness. We examine methods to quantify and understand this tradeoff at the algorithmic level.

3.1 Quantifying Consistency

ENetClus performs a clustering of the attribute type nodes, in which membership probabilities are assigned to nodes. Let the current data set being clustered belong to the time instant y and let the type we are interested in be t. Let the prior weight be fixed to λ_P, number of clusters be K and current level be l. Then, the membership probability of object o of type t to cluster c_i is denoted by $\{b_i(o)\}_{i=1}^K$.

Intuitively, consistency between two sets of cluster membership distributions is the degree of similarity between distributions of the intersecting objects. This implies that the insights derived from one set of the clusters should continue to hold valid over the next set, unless a major evolution has occurred. We can define consistency of the clustering c as the average cosine similarity between the cluster membership probability distributions of an object at time $y1$ and time $y2$. consistency(clustering c, $y1$, $y2$)= $\frac{1}{|O|}\sum_{o\in O}\frac{\sum_{k=1}^{K} b_k(o)_{y1} \times b_k(o)_{y2}}{\sqrt{\sum_{k=1}^{K} {b_k(o)_{y1}}^2}\sqrt{\sum_{k=1}^{K} {b_k(o)_{y2}}^2}}$

Such a comparison of consistency between two sets of clusters should be based only on the objects present in both time instants. Therefore, the set O used for the computation process denotes the set of intersecting objects at time $y1$ and time $y2$. Furthermore, we can define consistency for a particular level of the hierarchical clustering as the average consistency of sets of clusters at that level, each weighted by the number of objects in that set of clusters. Overall consistency is then an average of the consistencies at each level. Finally, we can express consistency across different types as a weighted sum of consistency with respect to each of the types.

The above definition defines the consistency only over *successive* snapshots. However, it can be easily generalized to the case of arbitrary intervals, by using the objects in the corresponding intervals. Furthermore, we can define a *chained path consistency* over a sequence of intervals as the product of consistencies over these corresponding intervals.

3.2 Quantifying Snapshot Clustering Quality

ENetClus represents each object o in a K-dimensional space when performing clustering. We could use the average ratio of intra-cluster similarity to inter-cluster similarity as a measure of the quality of a cluster. This is also called **compactness**. Higher values of compactness usually imply that the clustering is of better quality . The compactness C is defined as follows:

$C = \frac{1}{|O|}\sum_{k=1}^{K}\sum_{i=1}^{|O_k|}\frac{s(o_{ki},c_k)}{\sum_{k'\neq k} s(o_{ki},c_k')/(K-1)}$

where O is the set of the target objects, c_k is the centroid for cluster k and $s(a,b)$ measures the cosine similarity between K-dimensional vectors a and b.

We can define average **entropy** of a cluster as $E = -\frac{1}{|O|}\sum_{k=1}^{K}\sum_{o=1}^{|O_k|} b_k(o) \times log(b_k(o))$. Lower entropy means that on an average the objects belong to a particular cluster with high probability and so the clustering is of higher quality.

3.3 Cluster Merge and Splits

As the network evolves, different clusters can merge into a single cluster or a cluster can split into multiple clusters. Our soft clustering process has a fixed number of clusters for each time period. However, we can still study the merging and splitting of clusters as follows. Consider a set of clusters c at level l at time y. If at time $y+1$, a substantial part of the membership probabilities moves out of a cluster c_i to other clusters at the same level, then we can say that the cluster c_i has split into multiple parts. Similarly, if cluster c_i has obtained a substantial part of membership probabilities from other clusters at the same level at time $y-1$, then we can claim that cluster c_i has been formed from a merge of other clusters. While measuring the amount of merge or split of a cluster c_i, we should consider only those objects which occur in the network at both the times i.e., at y and $y-1$ when studying the "merge" phenomena and at y and $y+1$ when studying the "split" phenomena.

Continue rate: This is the rate at which the objects appear to continue in the cluster. If the membership probability of an object belonging to cluster c_i decreases, ratio of new probability to old probability is the continue rate of the object in cluster c_i. If the membership probability increases, the object is said to continue in the cluster with continue rate of 1.

Continue rate of cluster $c_i = \frac{1}{|O|}\sum_{o\in O} min(\frac{b_i(o)_y}{b_i(o)_{y-1}}, 1)$

where O is the set of the objects that occur in the network both at y and $y-1$.

Merge rate: This is the rate at which the objects appear to merge into a particular cluster c_i from all other clusters. An object is said to be merging into a cluster c_i if its cluster membership probability for c_i increases over time. If the

cluster membership probability decreases for the object wrt cluster c_i, it contributes 0 to the merge rate of cluster c_i.

Merge rate of cluster $c_i = \frac{1}{|O|}\sum_{o\in O} max(\frac{b_i(o)_y - b_i(o)_{y-1}}{b_i(o)_y}, 0)$

Split rate: This is the rate at which the objects appear to split out of a particular cluster c_i to all other clusters. An object is said to be splitting out from a cluster c_i if its cluster membership probability for c_i decreases over time. If the cluster membership probability increases for the object wrt cluster c_i, it contributes 0 to the split rate of cluster c_i.

Split rate of cluster $c_i = \frac{1}{|O|}\sum_{o\in O} max(\frac{b_i(o)_{y-1} - b_i(o)_y}{b_i(o)_{y-1}}, 0)$

Such rates can help us define interesting characteristics of the evolving data. E.g., in the case of bibliographic data, the merge and split rates can help us determine the influence of different research areas on another. The ability to determine both the evolutions of the clusters as well as the interactions of different parts of the network is key to the inference of interesting evolutionary insights in a multi-typed network.

3.4 Cluster Appearance and Disappearance

A cluster can be considered new, when most of its objects were not present in the previous time period. This can be formally defined in terms of the membership probabilities as follows: **Appearance rate** $= \frac{\sum_{o\in O'} b_c(o)_y}{\sum_{o\in O''} b_c(o)_y}$

Here, the set O' consists of objects which were not present at time $y-1$ and are present at time y. The set O'' consists of all objects in the cluster at time y.

Similarly, a cluster can be considered to be disappearing if most of the objects in it are absent at time $y+1$. **Disappearance rate** $= \frac{\sum_{o\in O'''} b_c(o)_y}{\sum_{o\in O''} b_c(o)_y}$

where the set O''' consists of objects which were present at time y and are not present at time $y+1$ and set O'' consists of all objects in the cluster at time y.

3.5 Stability of Objects

While many of the afore-mentioned definitions quantify the behavior of *clusters*, it is also interesting to quantify the evolutionary behavior of *objects*. The stability of an object quantifies the level to which the object is stable wrt its cluster or the network.

3.5.1 Temporal Stability

An object may appear continuously over multiple time instants or may appear intermittently. **Simple temporal stability** can be defined as the ratio of the number of time instants the object appears to the number of time instants in the observed time interval. **Sequential temporal stability** can be defined as the ratio of the number of time instants the object disappears to the number of time instants in the observed time interval. **Maximum sequential temporal stability** can be defined as the ratio of the maximum time interval for which the object is present in the network to the number of time instants in the observed time interval.

3.5.2 Simple Social Stability

The concept of social stability is based on cluster membership, and how frequently objects shift clusters. We define **simple social stability** as the ratio of number of times the object is retained in the same cluster to number of time instants it appears in the data. For soft clusters, we can assign every object to the cluster for which it has the maximum membership probability. We can also adapt the definition as the ratio of similarity between cluster membership distribution for the object over consecutive time instants to the number of times the object appears in the data.

3.5.3 Ranked Social Stability

The ranked social stability defines the level of stability among the *most representative objects* in the cluster. This is a natural metric for a clustering process which incorporates multi-typed ranking into the underlying algorithm. Let L_y and L_{y+1} be the list of the top k representative objects in a cluster at time y and $y+1$. **Ranked social stability** of cluster c can be formally evaluated by $\frac{|L_y \cap L_{y+1}|}{|L_y|}$. By varying the value of k, it is possible to gain deeper insights into the effects of the evolution on the most representative objects in the cluster. Since the most representative objects in the cluster can provide intuitive insights into the nature of the clusters, this provides a different perspective on how the clusters may have changed over time.

3.6 Sociability of Objects

Sociability of an object is the degree to which it interacts with different clusters. An object which belongs to many clusters is more sociable compared to one which belongs to a single cluster. It is a measure of the entropy of the membership of the object to different clusters. Therefore, social stability over a time interval I can be defined as $-\frac{\sum_i p_i \times log(p_i)}{log(K)}$ where K is the number of clusters and p_i is the ratio of number of times the object belongs to cluster i to the number of time instants in interval I for which the object was present in the data. For soft clusters, p_i can be defined as the ratio of sum of membership probability of the object in cluster i over I to the number of time instants in interval I for which the object was present in the data. This can often provide novel insights in a multi-typed network. E.g., in a bibliographic network, it can provide insights about how the terms or authors may evolve across different topical areas.

3.7 Effect of Social Influence

Multiple authors may move out of one research area to another. But there would still be some authors who may not. We need a metric to quantify the degree to which an object follows the cluster trend. Consider a vector V' which has size K^2. Let $V'(i,j)$ denote the movement of membership probabilities of an object o from cluster c_i to cluster c_j at time period $y+1$ i.e. the influence that cluster c_i has on cluster c_j via object o. Intuitively if cluster c_i influences c_j through object o, $b_i(o)_{y+1}$ should be less than $b_i(o)_y$ and $b_j(o)_{y+1}$ should be more than $b_j(o)_y$, otherwise V'(i,j)=0. In the first case, $V'(i,j)$ can be computed as $V'(i,j) = (b_i(o)_y - b_i(o)_{y+1}) \times (b_j(o)_{y+1} - b_j(o)_y)$. Finally, we normalize $V'(i,j)$ so that all elements add up to 1.

Consider another similar vector V of size K^2. We compute $V'(i,j)$ for every object $o \in O$ and store the average influence values in V. The cosine similarity between the vectors V and V' provides a good metric to measure the degree to which the object follows the trend. This can be called **normality**.

4. EXPERIMENTS

In this section, we will study the power of our tightly integrated approach of studying evolution and clustering in finding interesting cases in bibliographic networks. We use

the DBLP network. Such a network affords the ability to examine appearance of new research areas, cluster appearance rate for terms, authors and conferences. The goal of this section is to illustrate the power of our techniques in determining interesting changes in such a data set. Since the focus of this paper is to study clustering as a tool for evolutionary analysis of information networks, the focus of this section is also to make interesting observations about the underlying evolution with the use of such an approach.

4.1 Data Set

We perform clustering and study of evolution on DBLP data from 1993 to 2008. This data set contains approximately 654K papers, 484K authors, 107K title terms and 3900 conferences. The number of clusters was set to 4. We varied the prior weight between 0 and 1. The priors were specifically used for terms. The different node types in the graph were papers, authors, conferences and terms. Edges exist between paper and author node types, paper and conference node types and between paper and term node types. Our algorithm explicitly assigns paper nodes to particular partitions, and maintains a membership probability distribution for other nodes.

We also use a four area data set, which has been used earlier in [15]. This data set focuses on four information processing related areas. This is a subset of the DBLP data set containing approximately 24K papers (with 12K title terms) written by 26K authors in 20 conferences in the four areas of data mining, databases, information retrieval and machine learning over the years 1993 to 2008. We study the entire DBLP dataset in terms of slices and four area dataset in terms of snapshots.

4.2 Evolutionary Analysis of DBLP Data Set

We present some interesting results for the DBLP data set. Figure 2 shows that the #authors per paper has been increasing over time. We conjecture that this general trend in bibliographic networks is a result of greater collaborative efforts in recent years as a result of better communications and networking abilities, as well as better software support and enablement of collaborative efforts. We also observed that the #terms in the title of a paper has also increased over time. This is because the increasing complexity of research has brought in new terms into the vocabulary, while the old terms also continue to be used. With increasing maturity of research areas, authors have been writing more detail-oriented papers which need to use both the old terms and the new terms in order to describe the underlying topic.

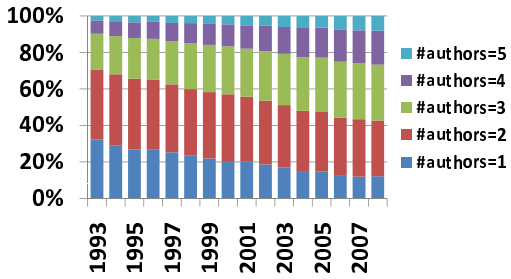

Figure 2: Evolution in the number of authors per paper

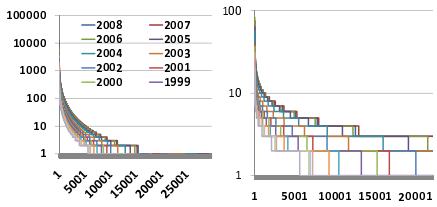

Figure 3: Power laws in the DBLP network: #papers vs. rank of a term(left), #papers vs. rank of an author(right)

Table 1: Consistency versus prior weight

Prior wt	0.0	0.2	0.4	0.6	0.8	1.0
Author	0.108	2.432	1.199	1.342	2.818	5.273
Term	0.470	3.105	2.168	2.222	3.646	6.024
Conf	0.567	2.730	1.800	1.326	3.293	6.709

We also studied the power law behavior of the bibliographic network. Figure 3 shows that the number of papers using a particular term in the title or the number of papers published by an author follow the expected power laws. One interesting observation was that the power law curve became more gentle over time. This is a result of the fact that the size of the network has increased over time. This suggests that larger bibliographic networks tend to have gentler power law curves.

4.3 Effect of Prior Weights

Table 1 shows the effect of varying the prior weight when performing clustering. The use of a higher prior-weight results in more consistency and smoothness in the clustering over different time-periods. We note that the consistency values rise as we increase the prior weight. The increase in prior weight increases the influence of the previous clustering on the current clustering. Introduction of new nodes in the network can result in fluctuations of the underlying consistency values. Thus, compared to the original NetClus, we achieve much more consistent clusters.

Table 2 shows the variation in compactness (defined in subsection 3.2) with increasing prior weight. Notice that the quality decreases initially and then improves as we increase the prior weight. The decrease happens because for lower prior weight, the clustering is confused between the prior information and the current information while when the prior weight is high, the clustering iterations align the current information around the prior information and tend to converge to a better local maxima of log likelihood. The prior information provide a firm initial clustering. This nature is quite different from other clustering algorithms where it has been shown that the snapshot quality decreases as the consistency increases. Also note that as prior weight is increased, the correspondence between two clusters between two snapshots increases. Hence, as against evolutionary K-means clustering, our clustering automatically results into matched clusters. Thus a prior weight of 0.8 helps provide both good consistency as well as good clustering quality. It would be interesting to see how quality and consistency change when priors are defined over other types like authors and conference also and at different time granularities. We

Table 2: Quality Variation with Prior Weight

Prior wt	0.0	0.2	0.4	0.6	0.8	1.0
Compactness	4594.175155	2166.869518	1978.39471	2932.194099	4267.585485	3972.629098

leave it as part of future work.

4.4 Continue, Merge and Split Rates

Figure 4 shows the continue, merge and split rates for different types of nodes where prior weight is fixed at 0.8. The rate values are averaged over all the years from 1993 to 2008. Also, in each histogram, each bar represents values for one level of the agglomerative clustering. Notice that the continue rate is more than merge and split rates. Also, merge rate is generally lower than the split rate. Looking out for outliers, we did observe a high split rate in DB cluster of 0.892 and a high merge rate of 0.8578 for DM cluster for authors in 2002 (denoting the rise of data mining from databases). There is a high merge rate for terms in IR cluster in time period 1998-1999 (possibly due to publications related to ranking techniques).

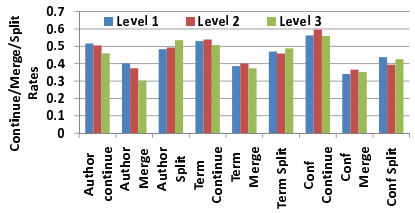

Figure 4: Variation of the merge, continue and split rates

4.5 Cluster Appearance and Disappearance

Figure 5 illustrates the appearance and disappearance rates for entire DBLP dataset for different types. The rate values are averaged over all the years from 1993 to 2008. A bar in each histogram represents values for one level of the agglomerative clustering. On average, the appearance and the disappearance rates increase as we go deeper into the lower levels of clustering, which represent finer grained topics of the bibliographic network. An intuition for this is as follows. Authors often publish in different sub-areas in different years, as a result of which they can appear to have disappeared from that sub-area in that time period. However, the author's major area usually remains the same, and hence the disappearance rate for authors would be higher in sub-areas than in major areas. This broad intuition is true across different kinds of evolution of the clustering process.

In the four area dataset, ML is the most dominant cluster in the first few years. We observe ML conferences at the top in DM and IR clusters for those years. But slowly in late 90s, we see IR and DM conferences appearing at the top.

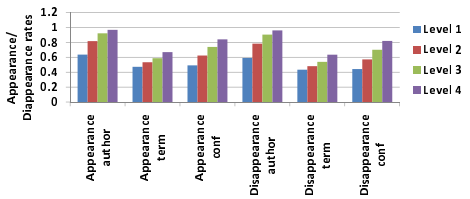

Figure 5: Variation of the appearance and disappearance rates

4.6 Evolution of Individual Nodes

While our afore-mentioned observations discuss the evolution of clusters, we will now study the evolution of individual nodes. We perform these experiments on the entire DBLP dataset. We study the evolution of individual nodes in terms of the stability metrics. Figure 6 shows the different types of temporal stability values in terms of the number of years. The figure shows the number of objects versus the temporal stability expressed in terms of number of years. Note that the conferences and terms are more stable than the authors. A stability value of 4 implies that the object disappeared from the graph 4 times in the 16 years in which it was represented. The trend for sequential stability is quite different compared to the simple and maximum sequential stability values.

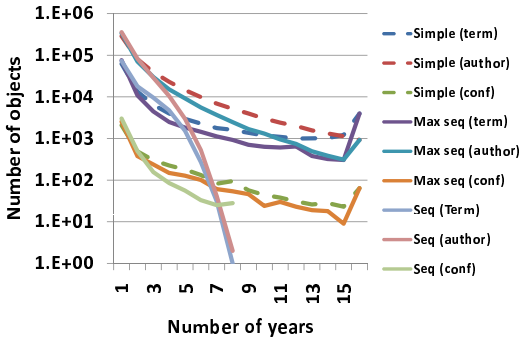

Figure 6: Temporal stability

Next, we plot the simple social stability values for the most temporally stable objects (i.e., objects which were present in our data for all the 16 years). Figure 7 shows that on an average objects maintain their cluster membership distribution upto a degree of 70%. The membership behavior of terms

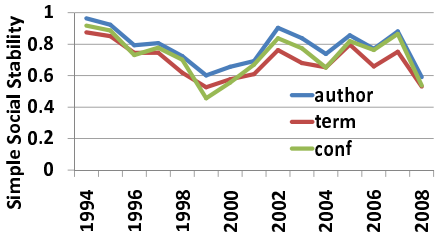

Figure 7: Simple social stability

and conferences in clusters is much more stable as compared to the membership behavior of authors. This is reasonable to expect, because the broad topics in the clusters evolve relatively slowly, whereas the authors may move in and out of different topical areas more rapidly. We note that such observations about the evolutionary behavior of information networks can be useful in order to identify the object types which show the most interesting evolution trends over time.

Further, we study ranked social stability for the nodes of type "term" with a prior weight of 0.8. The results are illustrated in Figure 8. The number of representative objects in the ranking was varied at top-$k = 10, 100, 1000$. While there is some variation in the results across different years, the results show that higher stability values are achieved by fixing k=10 as compared to k=100 or 1000. This suggests that only the most representative objects in the cluster continue to be stable, whereas the "modestly" representative objects may vary more significantly.

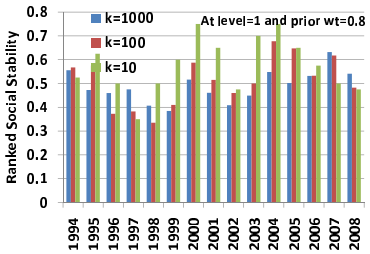

Figure 8: Ranked social stability

Figure 9 shows average social influence among different research areas using the four_area dataset across 16 years. Different bars represent different types. We can clearly see the influence between the DB and IR areas. We also notice ML to IR influence which is somewhat counter-intuitive. We think that this happens because in the first few years, since IR was not much developed, ML authors, conferences and terms occupy the "IR" cluster. Mutual influence between DM and ML is quite natural.

5. RELATED WORK

Traditionally, clustering has been performed using mincut, min-max cut, normalized cut, spectral and density-based methods in homogeneous graph networks. Sun et al. present a system called RankClus [14] and then NetClus [15] for clustering over heterogeneous information networks. We extended NetClus to perform agglomerative evolutionary clustering and then provided metrics to analyse these clusters

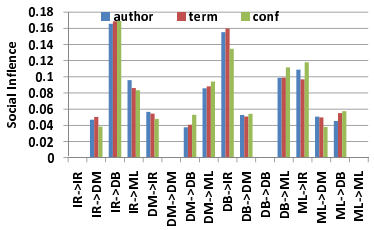

Figure 9: Influences among the four areas

and measure evolution. Our method could be extended by building one cluster tree sequence per type similar to [3, 11], which have different number of clusters per type.

Evolutionary clustering has been studied in some of the works [5, 4]. Chakrabarti et al. [4] proposed heuristic solutions to evolutionary hierarchical clustering problems and evolutionary k-means clustering problems. They introduce the concepts of consistency of clusters and cluster correspondence. Chi et al. [5] incorporate temporal smoothness in evolutionary spectral clustering which provides stable and consistent clustering results. They also handle the case when new data points are inserted and old ones are removed over time. While our framework automatically takes care of the new and old data points, we incorporate them separately when evaluating the similarity between clusterings. Also, unlike these works, we focus on evolution of heterogeneous networks. Mei et al. [12] discover and summarize the evolutionary patterns of themes in a text stream. Kumar et al. [7] study the evolution of structure within large online social networks. They present a segmentation of the network into three regions and study the evolution of these regions. The area of evolutionary clustering is also closely related to areas like clustering data streams. We leave storage and clustering of network data streams as future work.

Sun et al. [13] propose a system, GraphScope, which identifies communities in a parameter-free way, using the MDL principle. Kim and Han [6] perform evolutionary clustering using density-based methods. We use NetClus to identify clusters. Similar to their work, we can also track changes in clusters, appearance and disappearance of various clusters over time. Backstrom et al. [2] present an analysis of group formation and evolution in LiveJournal and DBLP. Some of our evolution metrics are influenced by their work. However, they define conferences in DBLP as clusters while we have typed-clusters obtained using NetClus. Leskovec et al. [8, 9] present a detailed study of network evolution. However, they do not deal with clustering of these graphs or study of the evolution of clusters. Tang et al. [16] study community evolution in a multi-mode network using a spectral framework. FacetNet [10] provides a framework for analysing communities and their evolution. We study evolution of clusters in much more detail. Apart from that the clusters obtained using the iterative NetClus algorithm have been shown to be more meaningful and hence studying their evolution is interesting. Asur et al. [1] characterize complex behavioral patterns of individuals and communities over time. They do not perform any temporally smoothed clustering.

6. CONCLUSION AND FUTURE WORK

In this paper, we designed a clustering algorithm for evolution diagnosis of heterogeneous information networks. This approach tightly integrates the evolution and clustering process, and provides novel insights into the evolution both at the object level and the clustering level. We studied the application of our approach on bibliographic information networks. We provided novel insights for evolution diagnosis on the DBLP data set, and showed the effectiveness of the evolution-sensitive clustering approach for heterogeneous information networks.

We can further modify the technique to incorporate variable number of clusters at different time periods. Also, it would be interesting to study the effect on compactness for different time granularities and when priors are defined for

other node types. Such an evolutionary clustering over heterogeneous information networks can also be helpful in identifying outliers in the network both in the static as well as evolutionary sense.

7. ACKNOWLEDGEMENTS

Research was sponsored in part by the U.S. National Science Foundation under grant IIS-09-05215, and by the Army Research Laboratory under Cooperative Agreement Number W911NF-09-2-0053 (NS-CTA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

8. REFERENCES

[1] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *KDD '07*

[2] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. *KDD '06*

[3] R. Bekkerman, R. El-Yaniv, and A. McCallum. Multi-way distributional clustering via pairwise interactions. *ICML '05*

[4] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. *KDD '06*

[5] Y. Chi, X. Song, D. Zhou, K. Hino, and B. Tseng. Evolutionary spectral clustering by incorporating temporal smoothness. *KDD '07*

[6] M. Kim and J. Han. A Particle-and-Density Based Evolutionary Clustering Method for Dynamic Networks. *VLDB 2009*

[7] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. *KDD '06*

[8] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. *KDD '08*

[9] J. Leskovec, J. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. *KDD '05*

[10] Y. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. Tseng. Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. *WWW '08*

[11] B. Long, Z. Zhang, X. Wú, and P. Yu. Spectral clustering for multi-type relational data. *ICML '06*

[12] Q. Mei and C. Zhai. Discovering evolutionary theme patterns from text: an exploration of temporal text mining. *KDD '05*

[13] J. Sun, C. Faloutsos, S. Papadimitriou, and P. Yu. Graphscope: parameter-free mining of large time-evolving graphs. *KDD '07*

[14] Y. Sun, J. Han, P. Zhao, Z. Yin, H. Cheng, and T. Wu. Rankclus: integrating clustering with ranking for heterogeneous information network analysis. *EDBT '09*

[15] Y. Sun, Y. Yu, and J. Han. Ranking-based clustering of heterogeneous information networks with star network schema. *KDD '09*

[16] L. Tang, H. Liu, J. Zhang, and Z. Nazeri. Community evolution in dynamic multi-mode networks. *KDD '08*