A Framework for Community Detection from Social Media

Thesis submitted in partial fulfillment
of the requirements for the degree of

MASTERS OF SCIENCE BY RESEARCH
in
COMPUTER SCIENCE

by

V Chandrashekar
200802038
`chandrasekhar.vug08@students.iiit.ac.in`

CENTER FOR VISUAL INFORMATION TECHNOLOGY
International Institute of Information Technology
Hyderabad - 500 032, INDIA
August 2013

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled "A Framework for Community Detection from Social Media" by V Chandrashekar, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date	Adviser: Prof. C. V. Jawahar
Date	Adviser: Dr. Shailesh Kumar

To My Family

Acknowledgments

I owe this work to the two men who mentored my work during my years at CVIT. First and foremost, I would like to express my deep-felt gratitude towards my research advisor Prof. C. V. Jawahar, whom I am indebted for life for his presence and keen interest in my research and able guidance. His ideas and support have not only been a constant source of motivation for me but have also been paramount in shaping my ideologies and understanding. He served both as an inspiration and a goading force throughout my stay at IIIT. Secondly, I was very fortunate to have a very involved advisor and mentor in Dr. Shailesh Kumar whose expertise in this field of research and valuable lessons in life kept me going through the tough times. This research would not have been possible without his continuous support and the interesting and insightful discussions I had with him.

I am thankful to my lab mates and CVIT peers who have been my friends, critics, mentors, pillars of support in times of distress, and a constant source of inspiration - Ankit, Jayguru, Aditya, Harshit, Mihir, Mayank, Abhinav, Vinay, Siddhartha, Yashaswi, Anand, Udit, Vidyadhar, Aseem and many more. Many thanks to Prof. P. J. Narayanan, Prof. Anoop and Prof Jayanthi for their encouraging presence and for providing an environment conducive to learning of the finest quality at CVIT. I am grateful to CVIT administration - Satya, Phani, Rajan for helping me in numerous occasions.

I would like to thanks all my friends at IIIT - Aman, Shashank, Nikhil, Rohit, Mohit, Omar, Abhilash, Piyush, Govind, Ayush, Jayant, Apoorv, Abhijeet, Harshit, Pranav, Siddharth, Abhishek, Nitesh, Ashish, Sagar, Jaspal, Jasmeet, Akshaymani, Tarun, and others for supporting me through good and bad times, and making life at IIIT a remarkable one.

Last but definitely above all, I would like to acknowledge the contribution of my parents to my academic and professional career. Their unconditional love, support and belief in my abilities have played a pivotal role in shaping the course of my career, and my life. I would like to particularly thank my late grandmother, who had always been my pillar of support during both my good and bad times.

Many thanks to everyone else who affected my life in any way, and wasn't personally acknowledged above.

Abstract

The past decade has witnessed the emergence of participatory Web and social media, bringing together people in many creative ways. Millions of users are playing, tagging, working, and socializing online, demonstrating new forms of collaboration, communication, and intelligence that were hardly imaginable just a short time ago. Social Media refers to interaction among people in which they create, share and exchange information and ideas in virtual communities and networks. Social Media also helps reshape business models, sway opinions and emotions, and opens up numerous possibilities to study human interaction and collective behavior in an unparalled scale.

In the study of complex networks, a network is said to have community structure if the nodes can be easily grouped into sets of nodes (even overlapping) such that each set of nodes is densely connected internally. Community structure are quite common in real networks. Social Networks often include community groups based on common location, interests, occupation etc. Metabolic Networks have communities based on functional groupings. Citation Networks form communities by research topic. Being able to identify these sub-structures within a network can provide insight into how network function and topology affect each other.

In this thesis, we design an end-to-end framework for identifying communities from raw, noisy social media data. The framework is composed of two important phases. First, we introduce a new method of converting the raw, noisy social media data into a weighted entity-entity co-occurrence based consistency network. This includes a simple iterative noise removal procedure for cleaning the entity consistency network by removing noisy entity pairs. Secondly, we propose an approach for identifying coherent communities from the weighted entity network, by introducing novel notions of communityness and community, based on eigenvector centrality.

We use this framework to solve three different problems from two distinct domains. The first problem involves detecting communities from raw social media data and showing the application of the communities discovered in a recommendation engine setting. We use the framework for converting the raw data into a clean network and propose a highly parallelizable seed based greedy algorithm to detect as many communities as possible from the weighted entity consistency network. Our framework for community detection is unsupervised, domain agnostic, noise robust, computationally efficient and can be used in different Web Mining applications like Recommendation Systems, Topic Detection, User Profiling etc. We also design an recommendation system to evaluate our framework with existing state-of-art frameworks [79, 23, 107] on a variety of large real-world social media data - Flickr, IMDB, Wikipedia, Bibsonomy, Medline. Our results outperform other frameworks by a huge margin.

The second problem is, given a set of communities of discovered by traditional community detection methods [76, 47], we need to identify loose communities among them and partition them into compact ones. Here, we use the second phase of our framework to identify such loose communities using our notion of community-ness and propose an algorithm for partitioning such loose communities into compact ones. We illustrate the results of our algorithm over Amazon Product and Flickr Tag data and compare its superiority over the traditional community detection methods in a recommendation engine setting.

The third problem is about showing the application of such framework in an Image Annotation scenario in presence of noisy labels. The problem of image annotation is defined to be, given an unknown image, we need to predict labels which best describes the semantics of the image. This problem is best solved in a supervised nearest neighbor setting, and we show how our framework can be used to address this problem, when the labels associated with training images can be noisy and redundant.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Networks are omnipresent on the Web. The most profound Web network is the Web itself comprising billions of pages as vertices and their hyperlinks to each other as edges [46]. Moreover, collecting and processing the input of Web users (e.g. queries, clicks) results in other forms of networks, such as the query graph [5]. Finally, the widespread use of Social Media applications, such as Bibsonomy[1], IMDB[2], Flickr[3] and YouTube[4], is responsible for the creation of even more networks, ranging from folksonomy networks [64] to rich media social networks [57]. Since networks originating from Social Media data are of particular interest to this study, we shall collectively refer to them as Social Media networks. Figure 1.1 shows example of different social media companies, which focus of different aspects of social media.

Despite the differences of Social Media networks with respect to the entities and the type of relations they model, they present a significant source of intelligence since they encode the online activities and inputs of masses of Social Media participants. Not only is it possible by analyzing such networks to gain insights into the social phenomena and processes that take place in our world, but one can also extract actionable knowledge that can be beneficial in several information management and retrieval tasks, such as online content navigation and recommendation. However, the analysis of such networks poses serious challenges to data mining methods, since these networks are almost invariably characterized by huge scales and a highly dynamic nature. Figure 1.2 shows the most popular social networks of the world as of December 2010. It could seen that most of globe is connected to each other by one or more these social media channels.

A valuable tool in the analysis of large complex networks is community detection. The problem that community detection attempts to solve is the identification of groups of vertices that are more densely connected to each other than to the rest of the network. Detecting and analyzing the community structure

[1]http://bibsonomy.org
[2]http://imdb.com
[3]http://flickr.com
[4]http://youtube.com

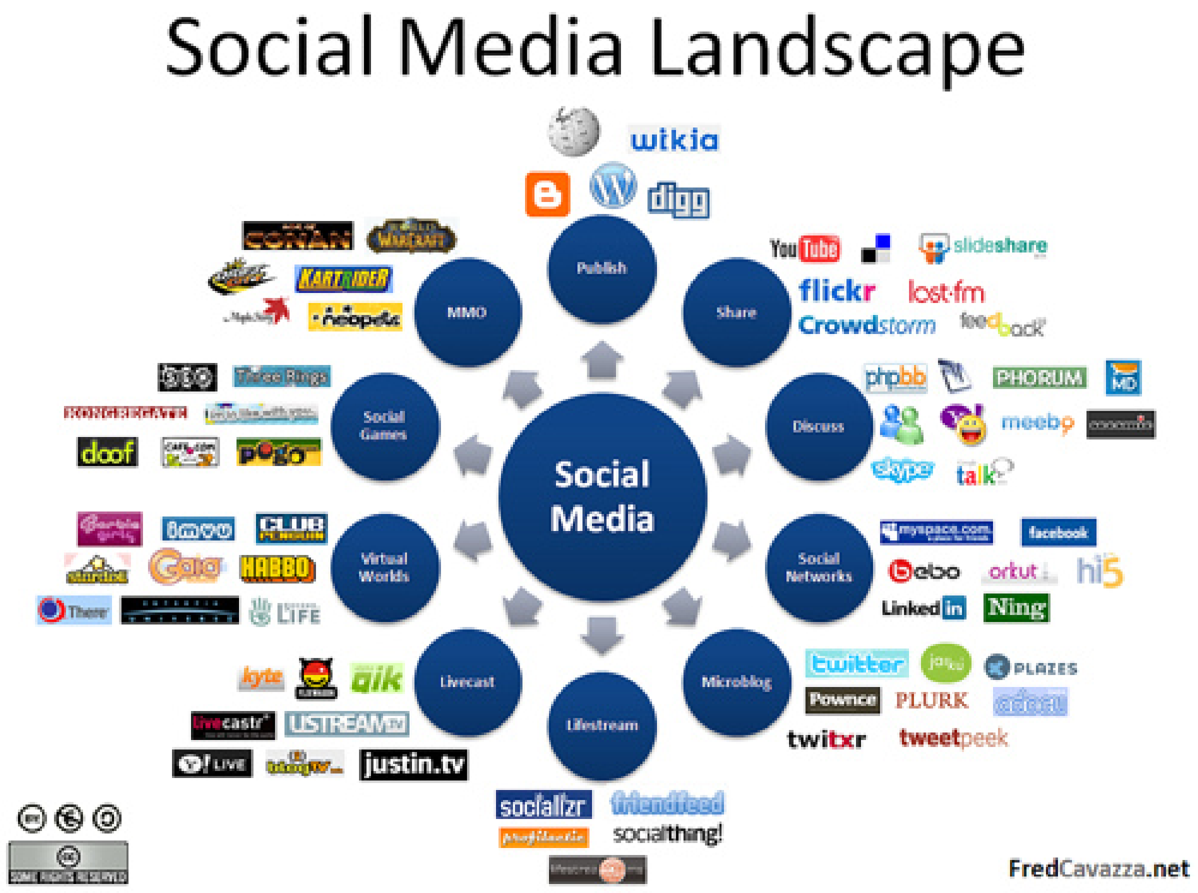


Figure 1.1 Social Media Landscape, showing the different aspects of social media.

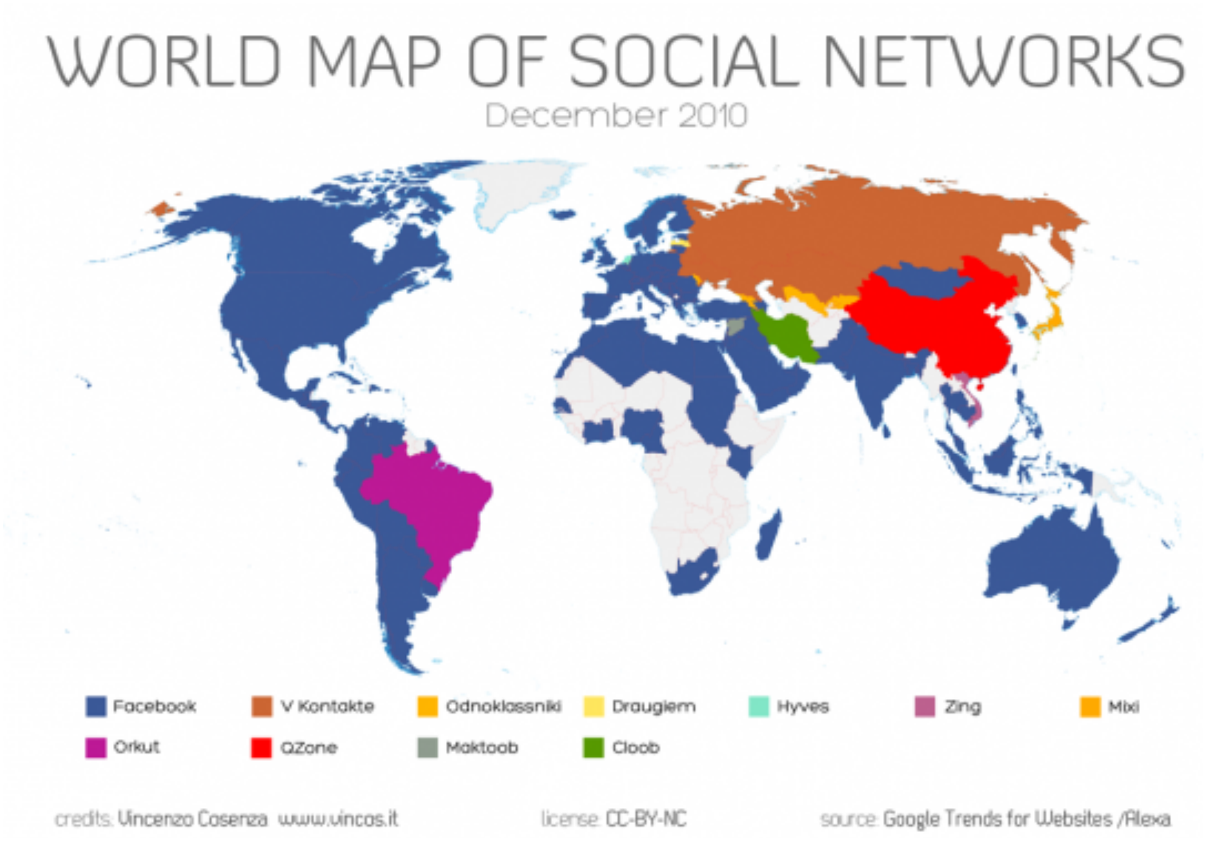

WORLD MAP OF SOCIAL NETWORKS
December 2010
Facebook
V Kontakte
Odnoklassniki
Draugiem
Hyves
Zing
Mixi
Orkut
QZone
Maktoob
Cloob
credits: Vincenzo Cosenza www.vincos.it
license: CC-BY-NC
source: Google Trends for Websites /Alexa

Figure 1.2 World Map showing the most popular social networks different parts of the world.

of networks has led to important findings in a wide range of domains, ranging from biology to social sciences [33] and the Web [46, 26]. Such studies have shown that communities constitute meaningful units of organization and that they provide new insights in the structure and function of the whole network under study. Recently, there has been increasing interest in applying community detection on Social Media networks not only as a means of understanding the underlying phenomena taking place in such systems, but also to exploit its results in a wide range of intelligent services and applications, e.g. recommendation engines, automatic event detection in Social Media content.

In this thesis, we present a robust end-to-end framework for mining coherent communities from social media data of different domains like Bibsonomy, Flickr, IMDB, Medline and Wikipedia. We illustrate the application of the communities discovered by our framework to tasks of recommendation and image annotation.

1.2 Challenges

The major challenges usually encountered in the problem of community detection in social media data are highlighted below:

- **Scalability**
  The amount of online social media content over the internet is rising everyday at a tremendous rate. Currently, the size of social networks are in scale of billions of nodes and connections. As the network is expanding, both the space requirement to store the network and time complexity to process the network would increase exponentially. This imposes a great challenge to the conventional community detection algorithms. Traditional community detection methods often deals with thousands of nodes or more.

- **Heterogeneity**
  Raw social media networks comprise multiple types of edges and vertices. Usually, they are represented as hypergraphs or k-partite graphs. Majority of community detection algorithms are not applicable to hypergraphs or k-partite graphs. For that reason, it is common practice to extract simplified network forms that depict partial aspects of the complex interactions of the original network.

- **Evolution**
  Due to highly dynamic nature of social media data, the evolving nature of network should be taken into account for network analysis applications. So far, the discussion on community detection has progressed under the silent assumption that the network under consideration is static. Time-awareness should be incorporated in the community detection approaches.

- **Evaluation**
  The lack of reliable ground-truth makes the evaluation extremely difficult. Currently the perfor-

mance of community detection methods is evaluated by manual inspection. Such anecdotal evaluation procedures require extensive manual effort, are non-comprehensive and limited to small networks.

- **Privacy**
  Privacy is a big concern in social media. Facebook, Google often appear in debates about privacy. Simple annoymization does not necessarily protect privacy. As private information is involved, a secure and trustable system is critical. Hence, lot of valuable information is not made available due to security concerns.

1.3 Key Contributions

The key contributions of this thesis are:

- Developing an end-to-end framework for detecting communities from noisy social media data. Over this process, we propose a simple noise removal procedure for cleaning raw social media network as well as introduce novel notion of community-ness and community in weighted networks.

- Designing a community-based recommendation system to show the superiority of our framework over other community detection methods.

- Building an algorithm for identifying loose communities discovered by traditional community detection methods and partitioning them into compact communities.

- Showing application of community detection framework in the task of image annotation.

1.4 Thesis Overview

The remainder of the thesis is organized as follows. In the next chapter, we give background on what social media is, how networks are created and also on the definition of community, and some of the recent works in community detection in social networks. We also talk about some application of communities in several social network analysis tasks. Chapter 3 presents our complete framework of mining communities from social media data. The framework contains two major modules: (i) First module describes the process of converting raw social media data into a weighted entity-entity co-occurrence consistency network. Here, we also introduce an iterative parametric noise removal procedure, which contributes as an important part of the framework. (ii) In the second module, we describe our mechanism of mining coherent communities from the weighted entity network. For this, we first introduce our measure of community-ness called coherence and our notion of community called Soft Maximal Clique, based on coherence. We also propose a greedy grow-shrink algorithm for exhaustively discovering almost all soft maximal cliques from the weighted network. Detailed evaluations of different modules of the complete framework is discussed in chapter 4, along with application to naive community-based entity recommendation on five different collaborative and expert tagging systems of different nature. In chapter 5, we propose an algorithm for compacting large and loose communities discovered by traditional community detection methods and show its effectiveness over Amazon and Flickr data. In chapter 6, we exclusively illustrate the application of coherence-based concept detection model in the task of image annotation in presence of noisy labels. Finally, in chapter 7 we draw conclusions from this thesis and also explore some of the possible avenues for future work.

Chapter 2

Background

2.1 Social Media

Social media refers to the means of interactions among people in which they create, share, and exchange information and ideas in virtual communities and networks. Andreas Kaplan and Michael Haenlein define social media as “a group of Internet-based applications that build on the ideological and technological foundations of Web 2.0, and that allow the creation and exchange of user-generated content.” [43]. Furthermore, social media depends on mobile and web-based technologies to create highly interactive platforms through which individuals and communities share, co-create, discuss, and modify user-generated content. It introduces substantial and pervasive changes to communication between organizations, communities, and individuals [45].

Social media differentiates from traditional/industrial media in many aspects such as quality, reach, frequency, usability, immediacy, and permanence. There are many effects that stem from internet usage. According to Nielsen, internet users continue to spend more time with social media sites than any other type of site. At the same time, the total time spent on social media in the U.S. across PC and mobile devices increased by 37 percent to 121 billion minutes in July 2012 compared to 88 billion minutes in July 2011 [72]. For content contributors, the benefits of participating in social media have gone beyond simply social sharing to building reputation and bringing in career opportunities and monetary income, as discussed in Tang, Gu, and Whinston [94].

2.1.1 Classification of Social Media

Social media technologies take on many different forms including magazines, Internet forums, weblogs, social blogs, microblogging, wikis, social networks, podcasts, photographs or pictures, video, rating and social bookmarking. Technologies include: blogs, picture-sharing, vlogs, wall-postings, music-sharing, crowdsourcing and voice over IP, to name a few. Many of these services can be integrated via social network aggregation platforms. By applying a set of theories in the field of media research (social presence, media richness) and social processes (self-presentation, self-disclosure) Kaplan and Haenlein created a classification scheme in their Business Horizons (2010) article, with six different types of social media: collaborative projects (for example, Wikipedia), blogs and microblogs (for example, Twitter), content communities (for example, YouTube and DailyMotion), social networking sites (for example, Facebook), virtual game worlds (e.g., World of Warcraft), and virtual social worlds (e.g. Second Life). However, the boundaries between the different types have been increasingly blurred. For example, Shi, Rui and Whinston (2013) argues that Twitter, as a combination of broadcasting service and social network, is better to be classified as a "social broadcasting technology." [88]

2.1.2 Elements of a Social Media Network

The ecosystem of Social Media applications comprises a wide range of objects that are associated to each other through numerous types of interactions and relations. Social Media networks provide an elegant representation of Social Media data, containing online objects as their vertices and the relations/interactions among them as edges. The vertices of Social Media networks can represent different types of actors, such as users, content items (e.g. blog posts, photos, videos), and even meta-data items (e.g. topic categories, tags). In addition, the edges of Social Media networks can be of different types, such as simple, weighted, directed and multi-way (i.e. connecting more than two entities) depending on the network creation process (discussed below).

In terms of notation, Social Media networks employ the typical graph notation $G = (V, E)$, where G stands for the whole network, V stands for the set of all vertices and E for the the set of all edges. Due to the different types of vertices and edges in such networks, it is common to consider sets of vertices and edges within V and E that contain vertices and edges of the same type. For instance, in the case of a photo sharing and tagging network, one can consider the set of vertices V to comprise the users, photos and tags of the system, i.e. $V = \{U, P, T\}$. Similarly, the set of edges in such an application would comprise the set of user-photo, photo-tag and user-tag associations, $E = \{UP, PT, UT\}$.

2.1.3 Social Media Network Creation

In practice, the creation of Social Media networks starts from a set of transactions that are performed and recorded in Social Media applications. Every such transaction typically involves different entities; for instance a tag assignment in Flickr involves a user, a photo and a tag, while a comment on a blog article involves the commenter, the blog article and the comment text. In that way, an association (edge) is formed between the items of the same transaction on an underlying network, so that the resulting Social Media network constitutes a direct representation of a subset of online transactions.

Since raw Social Media networks comprise multiple types of vertices and edges(some of which can be multi-way, i.e. link more than two vertices), they are mathematically represented by hypergraphs. Alternatively, if each hyper-edge of the graph is reduced to the pairwise connections among the k different types of nodes, the hypergraph is reduced to a k-partite graph. The majority of network analysis methods, and community detection in particular, are not applicable to hypergraphs or k-partite graphs.

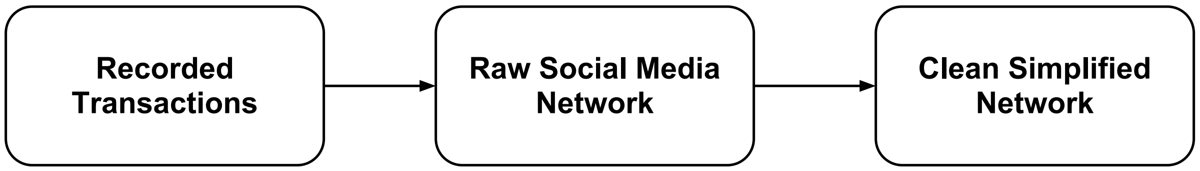


Figure 2.1 Typical lifecycle of a Social Media network: A set of transactions involving users, content and meta-data lead to the formation of a raw Social Media network. Typically, this network is simplified before any sophisticated analysis (e.g. community detection) takes place.

For that reason, it is common practice to extract simplified network forms that depict partial aspects of the complex interactions of the original network. Such networks are typically one- or two-mode (i.e. contain only one or two vertex types) and contain simple edges (i.e. connecting two vertices), which makes possible the application of numerous network analysis techniques. In summary, the typical lifecycle of a Social Media network (Figure 2.1) involves its creation from a set of recorded transactions and its transformation into some suitable form for the analysis that follows.

Folksonomies constitute an extensively studied example of Social Media networks. A folksonomy comprises three types of entities, namely users, resources and tags [64]. Starting from the tag assignments of users, i.e. transactions involving a user, a resource and a tag, a raw folksonomy network is formed, comprising three types of vertices and three-way relations among them. Subsequently, simpler (two-mode or one-mode) network representations are derived by use of projection operations [64, 84]. The raw tri-partite folksonomy network is transformed to a simple tag association network by considering an edge between two tags when they are used to tag the same resource. Other variants of deriving tag association networks from folksonomies are described in the works by Cattuto $et\ al.$ [16], and Yeung $et\ al.$ [109].

A more sophisticated paradigm for Social Media networks, termed meta-graph, is presented by Lin $et\ al.$ [57]. The vertices of a meta-graph are organized in facets and interactions among different facets, which can be multi-way, constitute the edges of the meta-graph. Another network model is proposed by Agichtein $et\ al.$ [1] for representing users, questions and answers in a community question-answering application. In conclusion, different forms of Social Media networks are possible depending on the transactions of the Social Media application under study, the modelling requirements of the problem, as well as the capabilities of the network analysis method at hand.

2.2 Communities

Due to the abundance of related works and the variety of adopted perspectives, there is no unique and widely accepted definition of community. Community definitions are formulated with reference to the network structure of the system under study and are commonly bound to some property either of some set of vertices (local definitions) or of the whole network (global definitions). However, at a different level, one should also define a community with respect to the domain under study, which in this survey comprises the realm of Social Media systems. For that reason, we will first provide a qualitative definition of Social Media communities and subsequently we will link this definition to established network-based definitions of quantitative nature.

At the most abstract level, given a Social Media network $G = (V, E)$, a Social Media community can be defined as a subgraph of the network comprising a set $V_C \subseteq V$ of Social Media entities that are associated with a common element of interest. This element can be as varied as a topic, a real-world person, a place, an event, an activity or a cause. For instance, in a blogging network, the set of all bloggers, articles, tags and comments related to the topic of renewable energy constitutes the respective community. Similarly, in a photo sharing application, the set of users, photos and tags that are associated with the island of Crete form a distinct community.

Social Media communities can further be described as explicit or implicit. Explicit communities are created as a result of human decision and acquire members based on human consent. Examples of explicit Social Media communities are Facebook and Flickr Groups. Implicit communities, on the other hand, are assumed to exist in the system and wait to be discovered. Implicit communities are particularly important for two reasons: (a) they do not require human effort and attention for their creation and (b) they enable the study of emerging phenomena within Social Media systems.

2.2.1 Community Structure and Attributes

Across all community definitions, a set-based view of communities is as follows: community was seen as a set of vertices and the membership of each vertex in a network was implicitly assumed to be the result of a boolean decision. In reality, and especially in the context of Social Media, the concept of community and community membership may be more complicated. For instance, in several of the above community definitions, e.g., most of the local ones [19, 60], the one based on Clique Percolation [76], and others [35, 17], it is possible for communities to overlap (Figure 2.2(a)). Community overlap is important for Social Media networks since it is common for Social Media entities to participate in multiple communities; for instance, a user may be affiliated to his/her family, friends and professional community.

In addition, there are other attributes that vertices of a network may have in relation to communities. For instance, different vertices may participate with varying degrees in a community depending on their centrality within it (Figure 2.2(b)). Moreover, vertices may have discrete roles: for example, Xu $et\ al.$ [104] define two roles (hubs and outliers) for vertices that are not assigned to any community. Hubs are connected to multiple communities and act as liaisons, thus enabling interactions among communities. Outliers are connected to a single community through a single link, therefore they are usually considered as noise. Community-based vertex roles are also discussed by Scripps $et\ al.$ [85]. Specifically, the roles roles of *loners*, *big fish*, *bridges* and *ambassadors* are defined (Figure 2.2(c)).

Finally, it is possible to impose hierarchical (Figure 2.2(d)) or multi-scale structures on communities. Community organization may be considered at different scales in a variety of systems. For instance, a

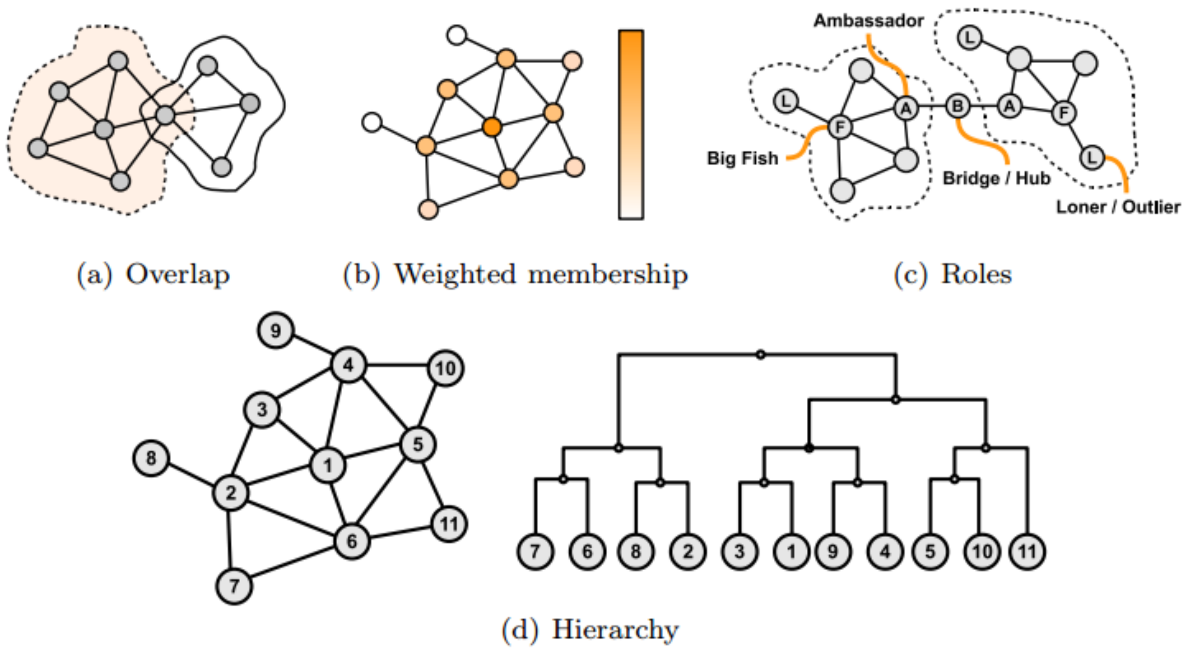


Figure 2.2 Several attributes that may characterize community structure: (a) overlap,(b) weighted membership, (c) vertex roles within/across communities,(d) hierarchical organization.

set of users of a Social Media application may be organized in a community focused on a very specific topic (e.g. fans of a particular indie-rock band) and at the same time they may be considered as members of a broader community (rock music). For Social Media systems, the consideration of multiple levels of community organization typically does not involve any kind of hierarchical organization since the constraints imposed by the hierarchical model are too restrictive for modelling the uncontrolled and emerging nature of Social Media phenomena.

2.2.2 Community Detection Methods

Community Detection is a broad field with many applications and approaches. An in-depth discussion on community detection can be found in Fortunato's work [27] and in a more recent survey by [102]. Various types of communities, communityness, and approaches for finding communities have been proposed in [78].

The key to a community detection algorithm is its definition of community-ness, the strength of a community. The main definitions of community-ness proposed in the literature can be broadly classified into four categories: (i) *Internal Community scores* such as Number of edges, Within Edge density, Average degree [80], and Intensity [74] (for weighted networks). They all use an aggregate property of the edges within the community. (ii) *External Score* such as Expansion [80], Cut Ratio [27], and betweenness centrality [33, 70]. These use an aggregate property of the edges connecting the community with the rest of the network, (iii) *Internal + External scores* such as Conductance [87], Normalized

Cut [87], etc. that combine properties of both the internal edges and external edges of the community, and (iv) *Network model* based scores such as Modularity [66] and its variants such as local modularity [18], and subgraph modularity [60]. These use a null hypothesis such as a random network with similar macro properties as the actual network. Most of these can be applied to both unweighted and weighted networks. Almost all of these measures have been built directly using edge weights and capture some property of the edge structure in the sub-network. Our community-ness measure *coherence*, is based on the relative importance of the nodes in the sub-network, which is calculated indirectly using edge-weights.

The state-of-art clique percolation algorithm (CPM) of Palla $et\ al.$ [76] finds overlapping communities by identifying the most densely connected parts. Farkas $et\ al.$ [23] extended CPM for weighted graphs by considering only those k-cliques, whose sub-graph intensity [74] exceeds a threshold during the percolation step.

Newman [66] introduced the notion of modularity as a measure of profoundness of community structure in a network. This spawned a whole class of *modularity maximization* methods for community detection [68, 19, 62, 21, 69]. A general problem with these methods is their tendency to produce clusters with a high skewed size distribution, leading to the conclusion that communities of small scales are likely to be undetected using modularity maximization [32].

Methods based on label propagation [81, 103], in which nodes with same label form a community, had been extended to overlapping community detection by allowing a node to have multiple labels. Local objective maximization based approaches like LFM [47], EAGLE [86], GCE [52], OSLOM [48], expands a community from a random seed node to form a natural community until the fitness function is locally maximal, also exist. Most of them rely on a local benefit function that characterizes the quality of densely connected group of nodes. his usually results in significant number of outliers or singleton communities.

Ahn $et\ al.$ [4] proposed clustering links instead of nodes using line graph of an undirected graph for community detection. They chose the jaccard index of the neighbourhoods of two nodes for analyzing links, which brought a whole new perspective for overlapping community study. Recently proposed community affiliation network models [105, 107] claim that the community overlaps are more densely connected than the non-overlapping parts and build models on bipartite node-community affiliation networks and can capture densely overlapping, non-overlapping and hierarchical nested communities in massive networks.

2.3 Applications of Communities

2.3.1 Topic Detection in Collaborative Tagging Systems

The huge amount of tags attached to online content by users of Social Media applications creates the need for imposing organization on the flat tag spaces of collaborative tagging applications. This can be directly achieved by grouping tags based on the topic they are associated with. There have been several recent works that attempt to derive meaningful clusterings of tags that correspond to topics of social interest. For instance, Begelman $et\ al.$ [6] were among the first to apply community detection methods, namely spectral modularity maximization, to identify interesting tag clusters. Similarly, tag clustering is pursued by means of a variant of the modularity maximization method of Newman [74] on enterprise folksonomies [89].

2.3.2 Tag Disambiguation

Due to the unrestricted and informal nature of tagging, there are numerous cases where the use of a single tag in isolation is not sufficient to convey the intended semantics. For that reason, tags need to be considered in context in order to disambiguate their meaning. Recently, several research efforts (Au Yeung $et\ al.$ [109], Specia and Motta [90]) attempted to address the problem of tag disambiguation by use of community detection. Starting from a particular tag, Au Yeung $et\ al.$ [109] derive several Social Media networks, e.g. a network of documents that have been tagged with the particular tag by the same user, and the community detection method of Newman [67] is applied to extract communities of tags or documents (that eventually lead to tags) that correspond to the different senses of a tag. This approach was demonstrated to yield superior performance compared to consulting some static external source of information such as WordNet.

2.3.3 User Profiling

Personalized search and recommendation constitute an additional information retrieval problem that can benefit from the use of community detection. More specifically, clusters of tags have been demonstrated to act as effective proxies of users' interests. Gemmell $et\ al.$ [30] base the ranking performance of a personalized search mechanism on tag clusters outperforming conventional ranking schemes. The tag clusters were extracted by use of a variant of hierarchical agglomerative clustering. However, since this scheme requires manual parameter tuning that may have significant impact on performance (as remarked by the authors), a viable alternative would be to use some community detection to identify tag clusters. Tsatsou $et\ al.$ [95] integrate the results of tag community detection in a personalized ad recommendation system and compared against conventional nearest-neighbor tag expansion schemes.

2.3.4 Event Detection

Events constitute an important unit of organization for Social Media content, since a large part of user contributed content revolves around real-world events. Community detection has found applications in the detection and tracking of events from social text streams. For instance, the framework presented by Zhao $et\ al.$ [111] incorporates textual, social and temporal aspects of blog feeds with the goal of tracking events. The N-cut graph partitioning method of Shi and Malik [87] is used twice in this framework: once to cluster a graph of blog posts connected by their textual similarity into topics, and at a second level, to cluster a graph of temporal activity profiles among users (created by their comments) into communities that correspond to real-world events.

Chapter 3

COOCMINER: Mining Coherent Communities in Entity-set Data

Entity-set data, such as market baskets in retail or tagsets in text, is growing at a tremendous rate in many domains. A retail market basket comprises of products purchased by a customer in a store visit. A tagset comprises of a set of keywords describing an object (e.g. YouTube video, Flickr image, or a movie, etc.). Entity-set data mining attempts to discover novel patterns, create actionable insights, engineer predictive features, and drive intelligent decisions from such data.

More than a decade ago, Frequent Itemset Mining (FISM) [2] powered by the *Apriori* algorithm [3] became the standard for finding *large* and *frequent* itemsets in entity-set data. As the vocabulary and data size grew, scaling the original Apriori algorithm became the primary focus of research. This lead to a number of innovations in scalable data structures and algorithms, some of which are FP-Growth [37], Eclat algorithm [110], Apriori by Borgelt [8], kDCI algorithm [59], and lcm [96]. Several other paradigms such as *rare itemset mining* [91, 92], *indirect association mining* [98], etc, emerged to address limitations of, and expand applications of the original FISM framework.

A common observation in traditional (direct) FISM is that it generates a very large number of noisy itemsets of which very few are really useful, novel, or actionable. In case of *indirect* association mining, where (potentially noisy) direct links are used to induce indirect associations, there is always a danger that the noise gets exaggerated and spurious indirect associations get created.

We start with the following definitions, observations and assumptions:

- Define **Logical concept** as a set of items that *completes* a *customer intent* in retail domain or *semantic concept* in a text or vision domain.

- These logical concepts are **latent** in the data and the goal of LISM is to **discover** them in a completely unsupervised fashion.

- The observed entity-set data may be best described as a **mixture-of, projections-of, latent logical concepts**, i.e. it has two fundamental properties: the mixture property and the projection property.

  - **Mixture property**: In retail, each market basket might contain *more than one customer intent*. Similarly, in text domain, each tagset might contain *more than one semantic concept*.

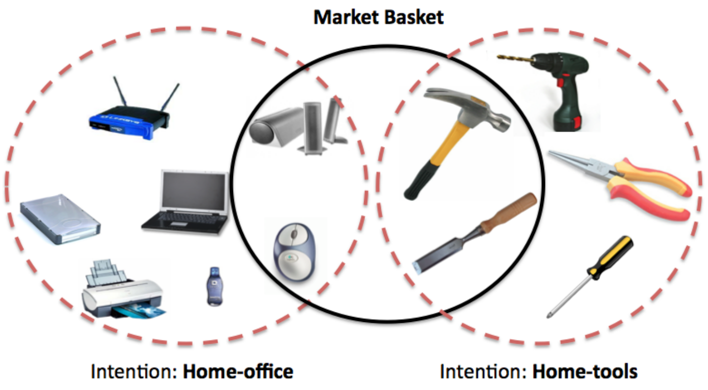


Figure 3.1 A hypothetical market basket (solid black circle) composed of items from two logical concepts (red dotted circles), representing *latent* customer intentions.

– **Projection property**: In retail, each market basket contains *only a subset of products* associated with a customer intent. Similarly, in text domain, each tagset contains *only a subset of keywords* associated with a semantic concept. In other words, a complete logical concept is rarely present in its entirety in the entity-set data.

Figure 3.1 shows a **hypothetical example** of the *mixture-of, projections-of, latent customer intents* in retail. Consider a market basket with four products (shown in solid black circle). These products come from two different customer intents, each represented by a *logical* group of products (shown in red dotted circles). In other words, (a) the market basket is composed of products from more than one customer intent (mixture property) and, (b) the market basket does not contain *all* products in either of the two intents (projection property). It only contains a subset of products associated with each intent. This could happen for various reasons: the customer already has the other items in those intents, she might purchase the remaining items in the intents elsewhere or at some other time, or she might not even be aware that the other items *complete* her intents, etc.

The *noise* due to the mixture property and the *incompleteness* due to the projection property make it challenging to discover the latent logical concepts from entity-set data. A complete logical concept will have a very low support as it hardly occurs in the data - thanks to the projection property. Also, each frequent itemset discovered by the traditional FISM framework might have sufficient noise in it - thanks to the mixture property. Finally, also note that some logical items might occur more rarely in the data than others. In fact in some cases it might be more useful to find rare itemsets [91] rather than frequent itemsets.

It should be fairly obvious from this discussion as to why frequent itemset framework is not a *natural* framework for discovering logical concepts and why we need a radically different framework for finding logical concepts in such data. It should also be obvious that unless we effectively deal with the mixture-of-intents noise, in the entity-set data, any indirect association mining will suffer from the propagation of this noise to higher order associations.

They are broadly two major approaches for finding **logical concepts** from entity-set data. One approach is the traditional topic modelling LDA [79], a very well known method in the text/web-mining community. But they may not be directly applicable to entity-set data for several reasons: (i) LDA is more suitable when the entity-set data is much larger as in bag-of-words in text or bag-of-visual-words in images, (ii) LDA depends on the *weight* (term frequency of a term in the document) of each entity in the entity-set but entity-sets inherently do not have such weights - an entity is either present or not present in the entity-set. (iii) The scaling and convergence properties of LDA will make it prohibitively costly to apply on such thin data, where the number of entity-sets is typically much larger than the size of each entity-set, and (iv) finally, LDA requires us to specify apriori the number of latent concepts to discover - something that is already hard in the text domain and is even harder in the retail domain.

The other approach is to convert the raw entity-set data into an entity-entity network and mine overlapping communities from this network. Community detection is a well studied paradigm and is used in a variety of domains such as physics, biology, computer science, social networks analysis, etc. It can be used to discover contextually or semantically related entities in co-occurrence data and borrows ideas from multiple disciplines such as information theory, graph theory, data mining, and social network analysis. Detecting communities [82, 27] in real-world networks is a hard and important problem, and has received a lot of attention.

Detecting communities [82, 27] in real-world networks is a hard and important problem, and has received a lot of attention. Real world networks contain overlapping communities where an entity (an ambiguous keyword in a tag network) might belong to multiple communities. Maximal cliques are a fundamental notion of tight overlapping communities in unweighted networks as they represent largest possible communities with the greatest possible edge density. Another consideration in community detection methods is their ability to work on weighted networks as the weights might contain significant information that should not be lost by thresholding them to unweighted networks.

Community-based approaches also face some issues while discovering concepts from entity-set data. First, the process of converting raw entity-set data into a robust entity network is not given enough importance. Usually, the entity-entity co-occurrences are directly used in the network generation process, and the projection and mixture properties of entity-set data is ignored. Second, while most community detection methods [87, 66, 81] can be naturally extended to weighted networks, there has been limited exploration of the notion of maximal clique on weighted graphs fundamentally, because cliques are defined only on unweighted graphs. Yang $et.\ al.$ [106], as part of their SNAP library, proposed some benchmark entity networks along with ground truth communities. But, these networks were unweighted.

In this thesis, we propose a simple and intuitive end-to-end framework called COOCMINER (Co-occurrence Miner) which addresses the mixture and projection properties in a novel fashion and discovers logical concepts from entity-set data by (a) systematically generating a noise-robust weighted entity co-occurrence network from possibly noisy annotated entity-set data, and (b) detecting communities in such weighted networks, using a novel definition of community, **Soft Maximal Cliques** (SMC). The SMCs discovered by COOCMINER constitute the logical concepts.

3.1 Generating Weighted Entity Consistency Network

COOCMINER first starts by first converting the raw entity-set data into robust entity network, something that most community detection methods do not give importance. It aggressively and systematically, removes noise in the network by (i) ignoring edges with low co-occurrence counts, (ii) normalizing edge weights by node priors to get more meaningful measures of co-occurrence strength between entities, and most importantly, (iii) applying a novel **iterative denoising step** that further cleans the entity co-occurrence noise.

3.1.1 Counts Statistics

Let $\mathbf{V} = \{v_m\}_{m=1}^{M}$ denote the vocabulary of all unique entities in the data. Let **X** denote the entity-set data with N data points: $\mathbf{X} = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(N)}\right\}$, where $\mathbf{x}^{(n)} = \{x_1^{(n)}, x_2^{(n)}, ..., x_{L_n}^{(n)}\}$ and L_n is the size of the n^{th} entity-set, $\mathbf{x}^{(n)}$. Three types of count statistics are computed from the entity-set data:

Co-occurrence counts defined over each pair of entities, $(\alpha, \beta) \in \mathbf{V} \times \mathbf{V}$ and denoted by $\psi(\alpha, \beta) = \psi(\beta, \alpha)$ is the number of entity-sets in which both entities "co-occur":

$$\psi(\alpha, \beta) = \sum_{n=1}^{N} \delta\left(\alpha \in \mathbf{x}^{(n)}\right) \delta\left(\beta \in \mathbf{x}^{(n)}\right), \tag{3.1}$$

(where $\delta(bool)$ is 1 if *bool* is true and 0 otherwise). The resulting $M \times M$ ($M = |\mathbf{V}|$ = vocabulary size) co-occurrence counts matrix, $\Psi = [\psi(\alpha, \beta)]$ is sparse and symmetric.

Marginal counts, $\psi(\alpha)$, for all unique entities $\alpha \in \mathbf{V}$ is defined as the number of co-occurrence pairs in which entity α occurred with some other entity in the entity-sets data. This is obtained simply by adding each row of the full co-occurrence counts matrix.

$$\psi(\alpha) = \sum_{\beta \in \mathbf{V}} \psi(\alpha, \beta). \tag{3.2}$$

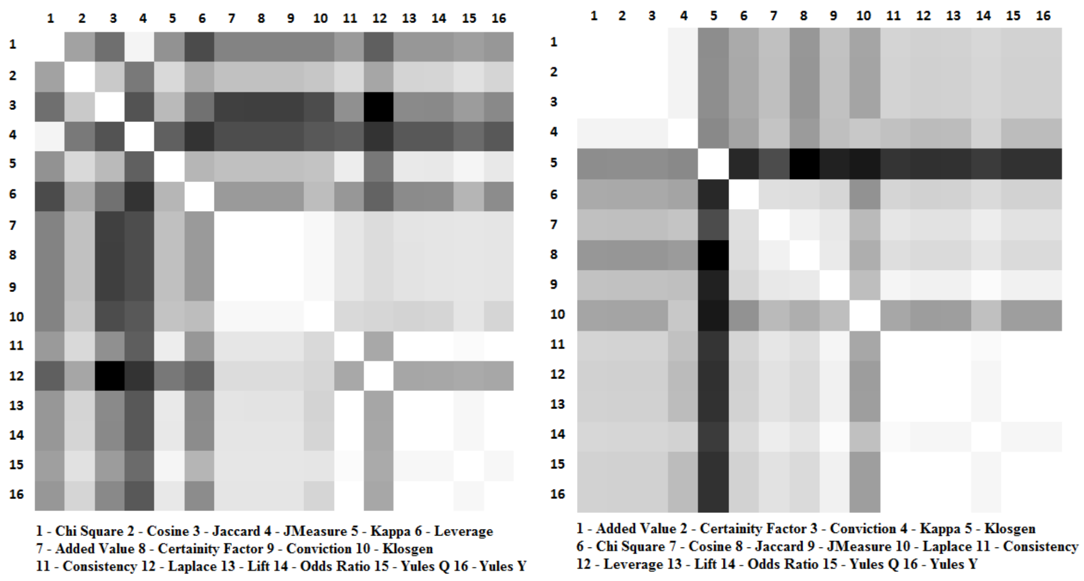
Total Counts, ψ_0 is defined as the total number of pairs in which some entity co-occurred with some other entity in the entity-set data. This is obtained by adding all the elements in the upper triangular co-occurrence counts matrix[1].

$$\psi_0 = \frac{1}{2} \sum_{\alpha \in \mathbf{V}} \psi(\alpha) = \frac{1}{2} \sum_{\alpha \in \mathbf{V}} \sum_{\beta \in \mathbf{V}} \psi(\alpha, \beta) \tag{3.3}$$

Co-occurrence and marginal probabilities are computed from these counts as[2]:

$$P(\alpha, \beta) = \frac{\psi(\alpha, \beta)}{\psi_0}, P(\alpha) = \frac{\psi(\alpha)}{\psi_0} \tag{3.4}$$

[1]Note that we divide this sum by 2 to only take the upper triangular part of the symmetrical matrix to avoid double counting.
[2]Smoothing may also be applied at this stage.



(a) Bibsonomy (b) Flickr

Figure 3.2 Correlation between 17 different consistency measures observed in Bibsonomy and Flickr datasets. The definitions of these measures can be found in [93, 31]. These measures were partitioned into 3 clusters based on their correlation.

3.1.2 Co-occurrence Consistency

The raw co-occurrence counts is not the best measure to quantify entity co-occurrence strength. It may happen that two very frequent entities (say `Chris` and `Eiffel tower`) might co-occur a lot compared to two relatively rare entities (say `global warming` and `green house emissions`) since frequent entities have a propensity to occur highly with other frequent entities just out of random chance rather than out of real association compared to a rare pair of entities. In order to discard the co-occurrence between `Chris` and `Eiffel tower` as noise and keep the co-occurrence between `global warming` and `green house emissions` as signal, we need to first transform their raw co-occurrence probabilities by normalizing with priors of the two entities in the pair. Instead of using raw co-occurrence counts, we use **co-occurrence consistency** measures that quantify how likely the entities are to co-occur in a entity-set vs. random chance. In other words, if the actual joint probability, $P(\alpha, \beta)$, is more compared to their random chance (e.g. $P(\alpha)P(\beta)$), then the two entities are said to have co-occurred with high consistency. This has precisely the effect we want - it gives high co-occurrence consistency between real (albeit low support) associations and reduces co-occurrence consistency between spurious (albeit high support) associations.

There are a number of measures that could be used here. We list the measures, which we have considered, as follows: **Added Value, Certainty Factor, Chi Square, Normalized Mutual Information, Conviction, Cosine, Jaccard, J-Measure, Kappa, Klosgen, Laplace, Leverage, Lift, Odds Ratio, Yules Q, Yules Y.** These definitions of all these measures can be found in [93, 31]. Using all these

Measure	Formula
AV	$max(P(B\|A) - P(B), P(A\|B) - P(A))$
NMI	$\frac{\log(\frac{P(A,B)}{P(A)P(B)})}{-\log P(A,B)}$
J	$\frac{P(A,B)}{P(A)+P(B)-P(A,B)}$

Table 3.1 Different Types of Consistency Measures for Association Patterns. AV = Added Value, NMI = Normalized Point-wise Mutual Information, J = Jaccard.

measures would be a tedious task, hence, we cluster the different consistency measures based on their correlation and pick one representative measure from each cluster for all further work. We found the Spearman Rank Correlation between the different measures and clustered the measures using METIS graph partitioning algorithm[44] into 3 clusters. Interestingly, we found the clusters to be similar for all datasets. Figure 3.2 shows an image representation of the correlation matrix, with the intensity representing correlation value (varying from black at the weakest correlation to white at strongest). We picked Added Value, Jaccard and Normalized Mutual Information to represent each of the three clusters respectively. Table 3.1 shows their definitions. We experimentally found **Normalized Point-wise Mutual Information** (NMI) [9] to be the most best performing (Section 4.3). NMI has some desirable properties like, it is in the range [-1, 1] and addresses the basic problem with another favourite measure, point-wise mutual information, which favors rare joint probabilities more.

We apply a threshold (θ_{consy}) on the co-occurrence consistency matrix also to remove the long tail low consistencies representing noise. Note that, unlike in other community detection methods, we do not binarize the graph, but keep all the co-occurrence consistencies, whenever they are above the threshold.

3.1.3 Noise Removal Procedure

Conversion of raw co-occurrence counts to co-occurrence consistencies does not guarantee that all noise is removed. In this section, we present an intuitive and novel iterative procedure to further denoise the co-occurrence consistency matrix. The intuition for this procedure is as follows: Consider a entity-set {`London, UK, Olympic, ceremony, bus, transportation`}. In the first pass through the data, there is insufficient knowledge to know whether a certain pair of entities in a entity-set is noise {`London, bus`} or signal {`bus, transportation`}. Hence, the initial co-occurrence between both noisy and signal pairs within each entity-set are counted equally. But, after computing co-occurrence consistencies between all pairs of entities, it is clear that certain entity pairs {`London, bus`} are noise because their co-occurrence consistencies are lower than θ_{consy} and certain other entity pairs {`bus, transportation`} are signal because their co-occurrence consistencies are higher than θ_{consy}. In the second pass, this knowledge can be use to ignore the noisy pairs within the entity-set (and not increment their co-occurrence counts) and only increment the co-occurrence counts of signal pairs of the entity-set.

Note that, multiple passes through the data are not required in this iterative method. The original co-occurrence counts matrix is masked by the co-occurrence consistency matrix in each iteration, new marginal and total counts are computed based on this masking, and the next co-occurrence consistency is reached. As denoising happens, mutual information [20] in the resulting consistency matrix improves. Mutual information [20] is measured in each denoising iteration k, as follows :

$$Q(\Phi^{(k)}) = \sum_{(\alpha,\beta)\in\mathbf{V}\times\mathbf{V}} P^{(k)}(\alpha,\beta)\phi^{(k)}(\alpha,\beta). \tag{3.5}$$

Figure 4.1 shows the effect of denoising on the co-occurrence consistencies across Bibsonomy, IMDB and Flickr datasets. A significant improvement was seen in mutual information of both the consistency matrix, measured by Equation (3.5), and also in the the final communities obtained after the denoising procedure. The complete entity consistency network generation process along with the noise removal procedure is described in Algorithm 1.

Algorithm 1 *NetworkGeneration*($[\psi(\alpha,\beta)]$)

```
 1: Iteration $t \leftarrow 0$
 2: $\psi^{(t)}(\alpha,\beta) \leftarrow \psi(\alpha,\beta)$
 3: $\psi^{(t)}(\alpha) \leftarrow \sum_{\beta\in\mathbf{V}} \psi^{(t)}(\alpha,\beta)$
 4: $\psi_0^{(t)} \leftarrow \frac{1}{2}\sum_{\alpha\in\mathbf{V}}\sum_{\beta\in\mathbf{V}} \psi^{(t)}(\alpha,\beta)$
 5: $P^{(t)}(\alpha,\beta) = \frac{\psi^{(t)}(\alpha,\beta)}{\psi_0^{(t)}}, P^{(t)}(\alpha) = \frac{\psi^{(t)}(\alpha)}{\psi_0^{(t)}}$
 6: $\phi^{(t)}(\alpha,\beta) \leftarrow$ Consistency($\psi^{(t)}(\alpha,\beta), \psi^{(t)}(\alpha), \psi^{(t)}(\beta),$
    $\psi_0^{(t)}$)
 7: $Q(\Phi^{(t)}) = \sum_{(\alpha,\beta)\in\mathbf{V}\times\mathbf{V}} P^{(t)}(\alpha,\beta)\phi^{(t)}(\alpha,\beta)$
 8: while $Q(\Phi^{(t)})$ converges do
 9:     $\psi^{(t+1)}(\alpha,\beta) \leftarrow \psi^{(t)}(\alpha,\beta)\delta\left(\phi^{(t)}(\alpha,\beta) > \theta_{consy}\right)$
10:     $\psi^{(t+1)}(\alpha) \leftarrow \sum_{\beta\in\mathbf{V}} \psi^{(t+1)}(\alpha,\beta)$
11:     $\psi_0^{(t+1)} \leftarrow \frac{1}{2}\sum_{\alpha\in\mathbf{V}}\sum_{\beta\in\mathbf{V}} \psi^{(t+1)}(\alpha,\beta)$
12:     $P^{(t+1)}(\alpha,\beta) = \frac{\psi^{(t+1)}(\alpha,\beta)}{\psi_0^{(t+1)}}, P^{(t+1)}(\alpha) = \frac{\psi^{(t+1)}(\alpha)}{\psi_0^{(t+1)}}$
13:     $\phi^{(t+1)}(\alpha,\beta) \leftarrow$ Consistency($\psi^{(t+1)}(\alpha,\beta), \psi^{(t+1)}(\alpha),$
        $\psi^{(t+1)}(\beta), \psi_0^{(t+1)}$)
14:     $Q(\Phi^{(t+1)}) = \sum_{(\alpha,\beta)\in\mathbf{V}\times\mathbf{V}} P^{(t+1)}(\alpha,\beta)\phi^{(t+1)}(\alpha,\beta)$
15:     $t \leftarrow t+1$
16: end while
```

Table 3.2 shows how the iterative procedure affects the consistency of tag `wedding` with some other tags in Flickr dataset. Note how its consistency with related tags such as `dress` & `reception` increases after the first iteration itself while it decreases to zero for unrelated tags, `chris` & `jason`. Table 3.3 shows examples of top related tags found for some random tags after the denoising procedure for both IMDB and Flickr datasets.

Tag	Before Denoising	After Denoising
bride	0.3257	0.5750
reception	0.3720	0.5728
marriage	0.3195	0.5658
cake	0.1699	0.3629
love	0.0148	0.2449
honeymoon	0.0183	0.2262
jason	0.2081	0
chris	0.1461	0

Table 3.2 Effect of noise removal on the consistencies of tags associated with tag **wedding** in Flickr dataset, with θ_{consy} = 0.001.

(a) IMDB

Tag	Most consistent tags
food	lifestyle, money, restaurant, drinking, cooking
road	truck, motorcycle, car, road-trip, bus
singer	singing, song, dancing, dancer, musician
suicide	suicide-attempt, hanging, depression, mental-illness

(b) Flickr

Tag	Most consistent tags
art	painting, gallery, paintings, sculpture, artist
france	paris, french, eiffeltower, tower, europe
island	tropical, islands, newzealand, thailand, sand
airplane	flying, airshow, fly, miltary, aviation

Table 3.3 Top 5 most consistent tags from the IMDB and Flickr datasets.

3.2 Community Detection in Weighted Entity Consistency Network

After the aggressive noise removal, it is expected that the final entity consistency graph $\Phi = [\phi(\alpha, \beta)]$ between all pairs of entities has high within community consistencies i.e. the higher the edge strength between a pair of entities, the more likely are they to belong to a community, and low across community consistencies i.e. two entities that are not connected strongly in it are most likely going to be part of different communities.

Maximal Cliques are a classical and well defined notion of tight overlapping communities on unweighted networks. But it cannot be used directly to find communities in weighted networks because cliques are defined on unweighted graphs only. One approach is to binarize the weighted graph by a threshold and then apply maximal cliques to find communities in it. This has several problems: first, it leads to a significant loss of edge weight information. Second, the binarization process is sensitive to the threshold. A small change in threshold could lead to a significantly different unweighted graph resulting in very different set of communities. Third, finding all maximal cliques is NP-hard [12].

The second stage of COOCMINER presents a novel definition of community called **Soft Maximal Cliques** (SMCs), which keep the weights intact and extends the notion of maximal cliques to weighted graphs. It major advantages are: no loss of edge weight information, no parameters, a more robust notion of community-ness based on a continuum of weights rather than the binary presence or absence of edges, and a scalable greedy algorithm that can be biased because of the weights, to explore the space of all combinations of nodes more systematically and efficiently to find communities.

At a high level SMC defines a new notion of community-ness called **Coherence** (Section 3.2.2) on a weighted subgraph. Unlike most community-ness measures that are defined directly in terms of edge weights [66, 74], coherence is defined in terms of node weights that we call **Local Node Centrality** (Section 3.2.1), which in-turn are computed from edge weights. SMCs are now defined as those subgraphs whose coherence is higher than all its "neighbors" (Section 3.2.3). We then present a seed based greedy grow-shrink algorithm for finding as many SMCs in the graph as possible through exhaustive seeding (Section 3.2.4).

3.2.1 Local Node Centrality

Most community-ness measures are direct aggregates over edge weights. For example, local density takes the arithmetic mean of all the edges or intensity [74] takes a geometric mean of all its edges, and modularity simply aggregates the difference between the actual and expected edge weight distribution [66]. Coherence, on the other hand, is defined indirectly. First we use the edge weights to derive *node weights* that capture how "important" a node is within a community. Then we aggregate these node weights into the coherence of the community.

To motivate node weights, consider the weighted network in Figure 3.3 and two subgraphs **A**= $\{$*rain, storm, cloudy, umbrella, chocolate*$\}$ and **B** = $\{$*candy, cocoa, chocolate, kid, milk*$\}$. It is pretty obvious that the entity *chocolate* "belongs" more in entity-set **B** and perhaps not at all in entity-set **A**. In other

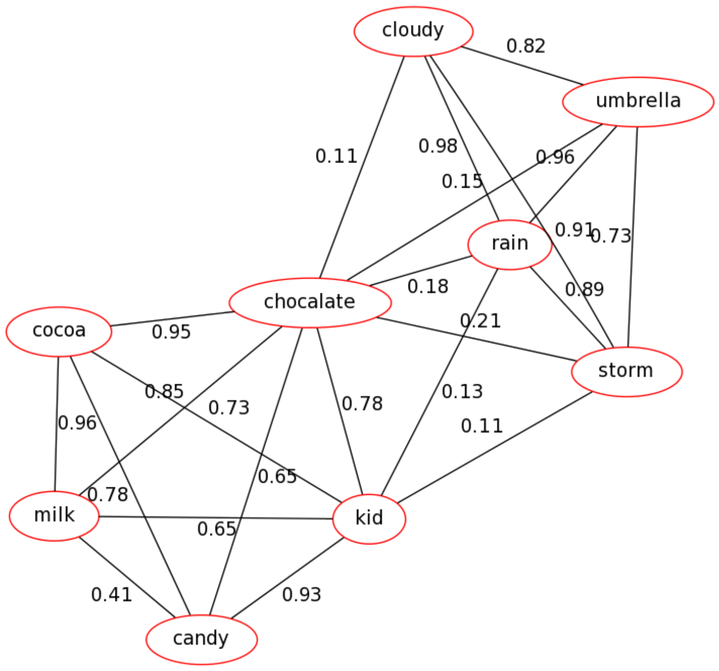


Figure 3.3 Example of a weighted network. Entity *chocolate* belongs more to {*candy, cocoa, chocolate, kid, milk*} than to {*rain, storm, cloudy, umbrella, chocolate*}.

words, if we were to assign a weight to each entity in the entity-set, we would assign a low weight to *chocolate* in **A** and a high weight in **B**. We do this because intuitively the weight of the entity should depend on the other entities it is present with. This intuition is captured in our node importance measure called **Local Node Centrality** (LNC) which is recursively defined as:

> A node is *central* to a community if it is *strongly connected* to other *central* nodes in the community.

Node centrality [75] is a well known concept in graph theory for capturing the global importance of a node in a (weighted) network. We adapt this abstract concept in three ways to define our local node centrality:

1. *Localization*: The first adaptation we do is to define node centrality locally w.r.t. each community, instead of globally for the entire graph since we need to capture the importance of a node within the community. So if a node belongs to multiple communities, it might get different centrality scores for each community - a desirable property.

2. *Eigenvector*: There are a number of measures of centrality to choose from: degree centrality, closeness centrality, betweenness centrality, eigenvector centrality [75]. We adopt eigenvector centrality which measures the influence of a node in the network such that, a node is more important if it is highly connected(in the weighted sense) to other influential nodes in the network. The

Dataset	Communities with LNC scores of entities
IMDB	**courtroom:0.92**, lawyer, trial, judge, perjury, lawsuit, false-accusation:0.53
IMDB	**africa:1.00**, lion, elephant, safari, jungle, chimpanzee, rescue:0.36
IMDB	**hospital:0.98**, doctor, nurse, wheelchair, ambulance, car-accident:0.43
Flickr	**wimbeldon:1.02**, lawn, tennis, net, court, watching, players:0.81
Flickr	**airplane:0.85**, plane, aircraft, flight, aviation, flying, fly:0.72
Flickr	**singer:0.84**, singing, musician, guitar, band, drums, music:0.72

Table 3.4 Examples of communities along with the LNCs of entities within the community, in order of centrality scores.

PageRank [11] is a variant of the eigenvector centrality. We use this for our purpose as it captures the essence of our definition perfectly.

3. *Unnormalization*: With the above two adaptations, the localized node centrality of a node in a community is the first eigenvector of its weight matrix. The L_2 normalization of eigenvectors is undesirable as it introduces community-size bias in a node's centrality scores - small communities will tend to get higher node centrality scores and large communities will tend to get smaller node centrality scores. To avoid this bias, we use unnormalized first eigenvector obtained by multiplying the first eigen-value to each element of the first eigenvector.

We now define LNC mathematically and tie it to the above definition. Let **x** = $\{x_1, x_2, ..., x_m\}$ be a set of m nodes in a subgraph and **W**(**x**) = $[w(x_i, x_j)]$ be the weight matrix associated with this subgraph, where $w(x_i, x_j)$ is the edge weight between nodes x_i and x_j. Let $\rho_t(x_i|\mathbf{W}(\mathbf{x}))$ denote the LNC of node x_i w.r.t. the subgraph in iteration t. Initialize all LNCs to be 1 (i.e. $\rho_0(x_i|\mathbf{W}(\mathbf{x}))$ = 1 $\forall i = 1...m$). Then, the LNCs are updated in each iteration until convergence:

$$\rho_{t+1}(x_i|\mathbf{W}(\mathbf{x}))) \leftarrow \frac{\sum_{j=1}^{m} \rho_t(x_j|\mathbf{W}(\mathbf{x})) \times w(x_i, x_j)}{\sqrt{\sum_{j=1}^{m} (\rho_t(x_j|\mathbf{W}(\mathbf{x})))^2}} \tag{3.6}$$

This converges to the first unnormalized eigenvector of **W**(**x**) i.e. if $\lambda_1(\mathbf{W}(\mathbf{x}))$ is the first eigen-value and $\mathbf{v}_1(\mathbf{W}(\mathbf{x}))$ is the first (normalized) eigenvector of this matrix then converged $\rho(\mathbf{x}|\mathbf{W}(\mathbf{x})) = \lambda_1(\mathbf{W}(\mathbf{x})) \times \mathbf{v}_1(\mathbf{W}(\mathbf{x}))$.

Table 3.4 shows few examples of communities found by SMC sorted by their centrality scores in entity networks. Note how, the earlier entities are more "central" to the community than latter entities.

3.2.2 Coherence of a Community

We propose a new heuristic measure called **coherence** loosely defined as follows:

A community is coherent if *each* of its nodes *belongs* with *all other nodes* in the community.

In other words, coherence is high if every node in the community has a high "belongingness" score. Even if one node is "peripheral" to it, the coherence goes down. According to our definition, all the nodes in the community must have a high LNC score in order for the community to have a high coherence score. So the most conservative definition of coherence would be to take the minimum of all LNC scores. This essentially makes sure that even if one of the nodes does not belong in the community, the community's coherence score goes down, no matter how high the LNC scores of other nodes in the community are. The "all-inclusive" nature of the minimum aggregation function resembles that of a clique in an unweighted graph. Therefore, minimum aggregated coherence score can also be referred to as *soft cliqueness* score of the community. Various aggregation functions in decreasing order of conservativeness (or cliqueness) are: minimum, harmonic mean, geometric mean, arithmetic mean, and maximum. Of these minimum is used by SMC as it matches the conservative nature of our definition of coherence. Mathematically, coherence is defined in terms of LNCs, as in Equation (3.7).

$$\pi(\mathbf{W}(\mathbf{x})) = \min_{i=1\ldots m} \{\rho(x_i|\mathbf{W}(\mathbf{x}))\} \tag{3.7}$$

Empirical evidence (Table 4.3) over these coherence measures support the intuition that using more conservative aggregation functions such as minimum, harmonic mean, and geometric mean gives better overall communities than the less conservative ones.

3.2.3 Soft Maximal Cliques (SMC)

We now describe the notion of *Soft Maximal Clique* in terms of these coherence values. Consider a network of four nodes: {*a, b, c, d*} with some weighted edges among them. We are interested in knowing which subgraph(s) of this graph i.e. which subset(s) of these four nodes (and their edges) are communities. Figure 3.4 shows a lattice representing the powerset of the four vertices, i.e. the set of all subsets of the vertices. Each element in this lattice is a subgraph comprising of those nodes (and implicitly all the edges among them) and is a potential candidate for a community depending on its coherence score and the coherence score of all its "neighbors". We first define neighborhood of a subgraph (or an element in the lattice) as follows.

> A subgraph **y** is said to be a neighbor of a subgraph **x** if **y** is obtained by either *removing* a single node from **x** or by *adding* a single node to **x**.

Let $\mathcal{N}(\mathbf{x}) = \mathcal{N}_+(\mathbf{x}) \cup \mathcal{N}_-(\mathbf{x})$ denote the neighborhood of subgraph **x** where $\mathcal{N}_+(\mathbf{x})$ denotes the *up-neighbors* obtained by *adding* a node currently not in **x** and $\mathcal{N}_-(\mathbf{x})$ are all the *down-neighbors* obtained by *removing* a node currently in **x** (note: **V** is the set of all nodes)[3]:

$$\begin{aligned}\mathcal{N}_+(\mathbf{x}) &= \{\mathbf{y} = v \oplus \mathbf{x}, \forall v \in \mathbf{V}\backslash\mathbf{x}\} \qquad (3.8)\\ \mathcal{N}_-(\mathbf{x}) &= \{\mathbf{y} = \mathbf{x}\backslash v, \forall v \in \mathbf{x}\} \qquad (3.9)\end{aligned}$$

Note that $|\mathcal{N}_+(\mathbf{x})| = |\mathbf{V}| - |\mathbf{x}|$ and $|\mathcal{N}_-(\mathbf{x})| = |\mathbf{x}|$, therefore, $|\mathcal{N}(\mathbf{x})| = |\mathbf{V}|$ for all $\mathbf{x} \in 2^{\mathbf{V}}$, the powerset of **V**. In the lattice structure (Figure 3.4), the *down-neighbor* of {*a, c, d*}, $\mathcal{N}_-(\{a, c, d\}) =$

[3]It is implicit that when a node is added or removed, all the relevant edges are also added or removed.

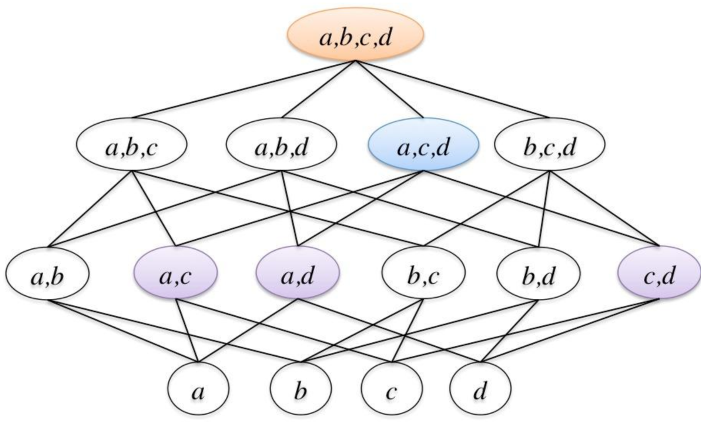


Figure 3.4 Entity-set {`a, c, d`} is a SMC, if the coherence of the entity-set {`a, c, d`} is higher than the coherence of all its *up-neighbors* ({{`a, b, c, d`}}) and all its *down-neighbors*, {{`a, c`}, {`a, d`}, {`c, d`}}.

$[\{a, c\}, \{a, d\}, \{c, d\}]$ as each of these subgraphs are obtained by removing exactly one node from {*a, c, d*} and its *up-neighbor* , $\mathcal{N}_+ (\{a, c, d\}) = [\{a, b, c, d\}]$ obtained by adding a node to it.

Let's assume that we have already computed the coherence of each subgraph or lattice element. In terms of coherence and neighborhood of a subgraph, we define a **Soft Maximal Clique** (SMC) as follows:

A subgraph $\mathbf{x}^*$ is a SMC if its *coherence* is higher than the *coherence* of all its *neighbors*.

More precisely, $\mathbf{x}^*$ is a SMC if: $\pi(\mathbf{x}^*) \geq \pi(\mathbf{y}), \ \forall \mathbf{y} \in \mathcal{N}(\mathbf{x}^*)$. All such subgraphs are considered communities. Note that [47] also defines neighborhood in a similar way but uses local maximization of modularity to find "natural" communities.

3.2.4 Grow-Shrink Greedy Algorithm

Clearly finding all SMCs in a weighted network is a computationally expensive task, as it would require that we evaluate each of the $O(2^{|\mathbf{V}|})$ subgraphs to see if they are SMC or not i.e. their coherence is higher than all their neighbors or not. Instead, in this section we present a systematic greedy algorithm that grows a community from a seed, does this for all possible seeds, and then removes duplicate communities. Unlike [47], however, we not just keep growing the community from the seed, we also allow the subgraph to shrink if that helps improve the coherence score of the subgraph. We now define the grow and shrink steps used in our greedy algorithm.

The **Grow** step finds the highest coherence *up-neighbor* of **x** in $\mathcal{N}_+(\mathbf{x})$. The basic grow step tries to exhaustively find the best node from $\mathbf{V} - \mathbf{x}$, to add to **x**. We reduce this complexity from $O(|\mathbf{V} - \mathbf{x}|)$ to $O\left(\left|\bigcap_{v \in \mathbf{x}} \mathbf{N}(v)\right|\right)$ by maintaining a **Grow Candidate List**, $\mathbf{C}(\mathbf{x}) = \{x' \in \mathbf{V} - \mathbf{x} | w(v, x') > 0 \, \forall v \in \mathbf{x}\}$ of only those nodes that are connected to *all* nodes in **x**, and search over this list for the best candidate.

Note that as the subgraph grows, its Grow Candidate List keeps shrinking. This leads to a significant speed-up without loss in quality.

The **Shrink** step finds the highest coherence *down-neighbor* of **x** in $\mathcal{N}_{-}(\mathbf{x})$. Again the basic shrink step tries to exhaustively find the best node to remove from **x**. We reduce this complexity from $O(|\mathbf{x}|)$ to $O(1)$ by picking the node in **x** with the least LNC, since removal of this node will maximally increase the coherence of the resulting down-neighbor. There is no guarantee that the most optimal node will be removed using this heuristic, but we found empirically that it is true more than 95% of the times.

A Soft Maximal Clique $\mathbf{x}^*$ is defined as the subgraph whose coherence is greater than both its **Grow** and **Shrink**:

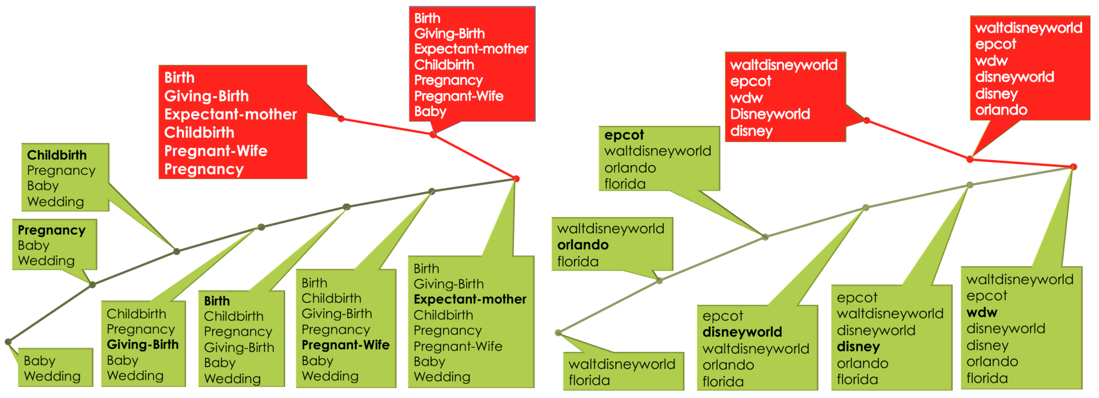
$$\pi(\mathbf{x}^*) \geq max\left\{\pi(\mathbf{x}^+ = Grow(\mathbf{x}^*)), \pi(\mathbf{x}^- = Shrink(\mathbf{x}^*))\right\} \tag{3.10}$$

The overall algorithm for finding soft maximal cliques, given an initial seed $\mathbf{x}_0$, is shown in Algorithm 2. It has two phases. In the **Grow Phase**, (lines 3-7 in Algorithm 2), we greedily keep adding the next node (and its relevant edges) until the coherence keeps increasing. This gives us the largest possible *Candidate Set*, starting from the original seed $\mathbf{x}_0$. The second stage is the **Grow-Shrink Phase**, (lines 9-24 in Algorithm 2) that can potentially alternate between a grow and a shrink depending on which step leads to the highest increase in the coherence value. The algorithm converges when neither a grow nor a shrink results in a subgraph with a higher coherence. Since, we want to find as many SMC communities as possible, we seed this with every pair of nodes $\mathbf{x}_0 = (\alpha, \beta)$ whose consistency $\phi(\alpha, \beta) > 0$. Also note that, multiple seeds might lead to the same community, so we finally remove all duplicate communities.

Figures 3.5 shows how the algorithm runs on a seed tagsets $\{baby, wedding\}$ and $\{waltdisneyworld, florida\}$ of IMDB and Flickr datasets respectively. Note that, due to the shrink stage, it is possible that the final community may not contain any or all nodes in the original seed used to create it.

Algorithm 2 SoftMaximalClique($\mathbf{x}_0$, Φ)

```
1:  $\mathbf{x} \leftarrow \mathbf{x}_0$
2:  A. **Grow Phase**
3:  $\mathbf{x}^+ \leftarrow Grow(\mathbf{x}|\Phi)$
4:  **while** $\pi(\mathbf{x}^+) > \pi(\mathbf{x})$ **do**
5:      $\mathbf{x} \leftarrow \mathbf{x}^+$
6:      $\mathbf{x}^+ \leftarrow Grow(\mathbf{x}|\Phi)$
7:  **end while**
8:  B. **Grow-Shrink Phase**
9:  **loop**
10:     $\mathbf{x}^+ \leftarrow Grow(\mathbf{x}|\Phi)$ {Best possible *up-neighbor.*}
11:     $\mathbf{x}^- \leftarrow Shrink(\mathbf{x}|\Phi)$ {Best possible *down-neighbor.*}
12:     **if** $\pi(\mathbf{x}^+) > \pi(\mathbf{x}^-)$ **then**
13:         $\mathbf{x}_{next} \leftarrow \mathbf{x}^+$ {Grow is better than shrink}
14:         $\pi_{next} \leftarrow \pi(\mathbf{x}^+)$
15:     **else**
16:         $\mathbf{x}_{next} \leftarrow \mathbf{x}^-$ {Shrink is better than grow}
17:         $\pi_{next} \leftarrow \pi(\mathbf{x}^-)$
18:     **end if**
19:     **if** $\pi_{next} > \pi(\mathbf{x})$ **then**
20:         $\mathbf{x} \leftarrow \mathbf{x}_{next}$ {Not reached local maxima yet.}
21:     **else**
22:         **return** $\mathbf{x}$ {Reached local maxima.}
23:     **end if**
24: **end loop**
```



Birth Giving-Birth Expectant-mother Childbirth Pregnant-Wife Pregnancy
Birth Giving-Birth Expectant-mother Childbirth Pregnancy Pregnant-Wife Baby
Childbirth Pregnancy Baby Wedding
Pregnancy Baby Wedding
Baby Wedding
Childbirth Pregnancy Giving-Birth Baby Wedding
Birth Childbirth Pregnancy Giving-Birth Baby Wedding
Birth Childbirth Giving-Birth Pregnancy Pregnant-Wife Baby Wedding
Birth Giving-Birth Expectant-mother Childbirth Pregnancy Pregnant-Wife Baby Wedding
waltdisneyworld epcot wdw Disneyworld disney
waltdisneyworld epcot wdw disneyworld disney orlando
epcot waltdisneyworld orlando florida
waltdisneyworld orlando florida
waltdisneyworld florida
epcot disneyworld waltdisneyworld orlando florida
epcot waltdisneyworld disneyworld disney orlando florida
waltdisneyworld epcot wdw disneyworld disney orlando florida

(a) IMDB (b) Flickr

Figure 3.5 Actual example runs of grow-shrink algorithm starting with an edge seed in IMDB and Flickr data. Tags within a sub-graph are sorted in descending order of their LNC scores. x-axis is the size of the tagset and y-axis is the **coherence** of the sub-graph. Each grow step (green) shows tags highlighting (bold) the new tag added. Each shrink step removes the last tag in the previous iteration.

Chapter 4

Experimental Evaluation

We study the effectiveness of our framework in two parts: (a) by evaluating the importance of noise removal step during the network generation phase and, (b) by comparing the quality of the communities obtained by SMC with two popular algorithms, Weighted CPM (WCPM) [23] and BIGCLAM [107]. We start by describing the datasets and the metrics used for evaluation. Then, we examine the importance of noise removal step in the process of generating weighted entity networks from raw entity-set data. To understand the significance of our SMC algorithm for the task of community detection, we do a comparative study of the structural properties of the communities discovered by SMC and two other popular algorithms and show the effectiveness of SMC over other methods in an application-oriented community-based entity recommendation task. Finally, we analyse the role of parameter θ_{consy} in the performance of the complete framework.

4.1 Datasets

Weighted networks were generated from five real world collaborative tagging systems where an object (e.g. an image or movie) is annotated by a set of entities/tags.

1. BIBSONOMY[1]: tags for 40K bookmarks and publications.
2. FLICKR[2]: collection of 2 million social-tagged images randomly collected from Flickr.
3. IMDB[3]: Keywords associated with about 300K movies.
4. MEDLINE[4]: containing references and abstracts on about 14 million life sciences and biomedical topics. We use the Mesh terms associated with the topics as entities.
5. WIKIPEDIA: where the wikipedia pages were treated as entities and the out-links of a page were used to create entity-set of that page. We used around 1.8 million pages to create our dataset.

[1]ECML-PKDD Discovery Challenge 2009
[2]http://staff.science.uva.nl/ xirong/index.php?n=DataSet.Flickr3m
[3]www.imdb.com
[4]http://www.ncbi.nlm.nih.gov/pubmed

	T	N	D(%)
BIBSONOMY	43,114	12,215	0.262
IMDB	280,833	5,000	21.42
WIKIPEDIA	1,782,857	5,000	31.00
FLICKR	2,134,760	5,000	18.56
MEDLINE	14,036,218	5,000	30.49

Table 4.1 Properties of the weighted networks generated using the collaborative tagging datasets. T, N and D denotes the Number of Entity-sets, Number of Nodes (Entities) and Initial Network Density respectively.

Each node in the network is a entity and the weight between two nodes quantifies how likely the two entities will co-occur relative to random chance. The process of generating weighted networks from these entity-sets is described in chapter 3 section 3.1. Table 4.1 shows some basic statistics i.e. the number of entity-sets, number of unique entities used, and entity network density of the initial networks (before noise removal).

4.2 Evaluation Metrics

Like in any other unsupervised learning task it is not obvious how to objectively evaluate and compare the quality of communities obtained by various methods. In this chapter, we use two previously proposed methods for such evaluations: *Modularity* - a direct, unsupervised metric and *Community based Entity Prediction* - an indirect, supervised metric for evaluating communities.

Overlapping Modularity [71] extends the classical notion of modularity [66] defined for *non-overlapping communities* to *overlapping communities*, by introducing notion of belonging coefficients. For our experiments, we have used the weighted version of this modularity and set the belonging coefficients of a vertex v to $\frac{1}{c(v)}$, where $c(v)$ is number of communities the vertex is participating. Even though modularity maximization is used as a criteria for evaluating many community detection methods, it need not always coincide with the correct or best communities, as pointed in [34]. What we want to evaluate is not how well a graph measure is maximized, but how good is our community discovery algorithm in describing real world knowledge about the entities being grouped. Therefore we use another measure also to evaluate communities. Any further use of the term modularity in this thesis refers to the overlapping modularity definition.

Community based Entity Prediction [77] uses a simple entity prediction system built on top of communities to objectively evaluate their quality. First, split the entire entity-set data into training and test sets. Second, use training set to build the weighted network and find communities in it using various methods. Third, build a prediction system using these communities, and finally evaluate the quality of the prediction system on the test set. The basic prediction system, via communities, works as follows: (i) From each test entity-set, remove all other entities, except 1. The goal is to see how well we predict the removed (target) entities from the remaining (input) entity, via communities. (ii) Find all the

	Bibsonomy			Flickr		
	NMI	AV	J	NMI	AV	J
P(%)	18.04	17.15	17.71	13.28	12.19	12.82
R(%)	14.48	8.4	6.8	11.96	8.14	6.47
F(%)	16.07	11.27	9.82	12.59	9.76	8.6

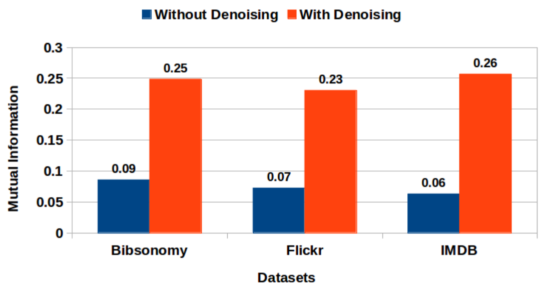
Table 4.2 IR performance of communities discovered using Normalized Mutual Information (NMI), Added Value (AV) and Jaccard (J) as consistency measures in entity recommendation. P, R, F denotes Precision, Recall and F-measure respectively.

communities that contain the input entity. Lets call these the *recommended communities*. (iii) Take the union of all entities in these *recommended communities*. These form the *predicted entities*. Score each predicted entity by the number of *recommended communities* in which it is present. Lets call this the *recommendation score* for each predicted entity. (iv) Sort all predicted entities by this *recommendation score* and find the ranks of the target entities in this list. (v) Evaluations were done by calculating precision, recall and f-measure. Precision is defined as the fraction of correct predictions relative to the total number of predictions made. Recall is defined as the fraction of correction predictions relative to the total number of entities in the ground-truth. F-measure is the harmonic mean of Precision and Recall.

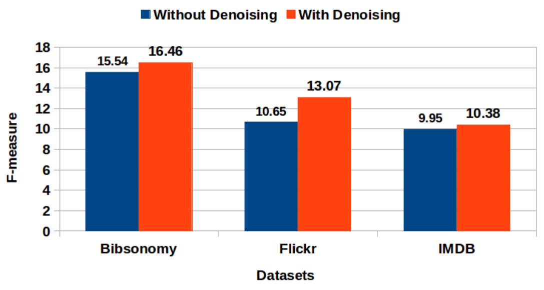
We used 70-30 training test splits with 5-fold cross-validation. Also, the test set was cleaned up as follows: First, top 5% of the entities were removed otherwise predicting them itself would give high performance. Second, entities not present in the training set were removed from the test set also.

4.3 Study of Different Consistency Measures

In Section 3.1.2, we had talked about 16 different consistency measures, which can be used to model pairwise relationships between entities in the data. For each dataset, we found correlation between these measures and clustered them into three groups. We used one measure from each of these clusters, to represent each cluster. We used the following three interestingness measures : NMI, Added Value and Jaccard. Since, qualitative comparison of the communities discovered, using each of the three measures, would be inappreciable, we do quantitative comparison of the communities discovered, by the significance of their IR performance in entity recommendation. Table 4.2 compares the Precision, Recall and F-measure of each of the three consistency measures for entity recommendation in Bibsonomy and Flickr datasets. Recall using Added value and Jaccard is low compared using NMI, whereas precision is about the same using these measures.



(a) Mutual Information



(b) F-measure

Figure 4.1 Effect of noise-removal procedure in weighted entity network generation phase across Bibsonomy, Flickr and IMDB datasets. We compare networks generated without denoising and networks generated after applying denoising. Evaluations have been done over (a) Mutual Information [20] of the network, (b) the F-measure in entity recommendation. For noise-removal θ_{consy} = 0.001 was used and communities were discovered using SMC.

4.4 Effect of Noise Removal in Entity Network Generation Phase

Most entity networks use pair-wise raw entity co-occurrence counts to exemplify the relationships between entities. However, due to deceptive high co-occurrence counts between noisy entity-pairs and low co-occurrences between rare entity-pairs, consistency-based measures [93, 31] have been used instead of raw counts. Usually, this consistency-based entity network is thresholded to remove spurious entity-pairs and the remaining weighted entity network is used for community detection. However, a threshold-based approach is not the most efficient way to remove noisy links. We propose an iterative noise removal procedure to remove such spurious links, described in Section 3.1.3.

To understand the significance of our noise removal procedure, we compare the consistency-thresholded networks and the consistency networks after applying noise removal step, using (a) direct measure : Mutual Information [20] of the consistency networks and, (b) indirect measure : Entity Prediction performance of the communities derived from the networks. We used θ_{consy} = 0.001 for noise-removal procedure and SMC for community detection.

Figure 4.1 illustrates the experimental findings on Bibsonomy, Flickr and IMDB datasets. For Bibsonomy, there is 177.78% increase in mutual information of the networks after applying noise-removal, whereas for networks like Flickr and IMDB, there is significantly high 228.57% and 333.33% increase respectively. To understand the impact of noise removal over the complete framework, we compare the performance of the two networks in the task of community-based entity recommendation. For Bibsonomy and IMDB, there is 5.92% and 4.32% increase in F-measure, whereas for user-collaborative network Flickr, there is exceptionally high increase of 22.72% in F-measure. In any case, our noise-removal procedure doesn't deteriorate the performance of the framework, rather tries to improve its effectiveness, wherever possible as in case of Flickr.

	Min	HMean	GMean	AMean	Max
N	7359	7228	7250	7168	7083
S	4.69	4.21	4.13	4.12	3.88
M	**0.54**	0.56	0.55	0.50	0.47
F	**16.07**	15.51	15.43	15.39	15.29

Table 4.3 Comparison of different functions of aggregating Local Node Centralities into Coherence on Bibsonomy Dataset. N = number of communities, S = average community size, M = modularity [23], and F = F-measure on entity recommendation

.

4.5 Aggregation Function for Soft Maximal Cliques Coherence

In section 3.2.2, coherence of a community was defined as an aggregation of the Local Node Centrality (LNC) of all nodes in it. We argued that using a conservative aggregation function such as minimum or harmonic mean better captures the notion of community compared to more loose aggregation functions such as maximum or arithmetic mean. Table 4.3 compares different aggregation functions in terms of both Modularity and F-measure on Community based Entity Prediction metrics on Bibsonomy dataset (results on all datasets were consistent). As the aggregation function becomes less conservative from minimum, to harmonic mean to geometric mean, to arithmetic mean, to maximum, the quality of communities detected decreases - Minimum performs the best and the Maximum performs the worst. Both the number and size of communities also increases as the coherence measure becomes less conservative. Hence, we used Minimum of the LNCs as our definition of Coherence.

4.6 Comparing SMC with other Community Detection Methods

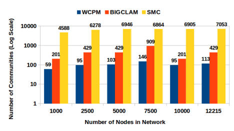
Here, we compare the communities discovered by SMC with two other popular overlapping community detection algorithms over all 5 datasets, namely Bibsonomy, Flickr, IMDB, Medline and Wikipedia.

- *Weighted Clique Percolation Method* (WCPM) [5] [23], an extension of the state-of-art CPM [76] for weighted networks, based on the concept of percolating k-cliques with high intensity [74]. The algorithm allows overlaps between the communities. In our experiments, we tried different values of the percolation factor (k) and reported results corresponding to the best value of k.

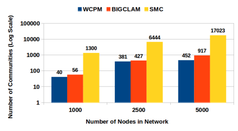
- *BIGCLAM* [6] [107], a recently proposed fast overlapping community detection method based on non-negative matrix factorization [53, 40, 56], for undirected networks. It is built on the model of affiliation networks [10, 49] and communities arise due to shared community affiliations of nodes.

[5]Implementation available at http://cfinder.org/

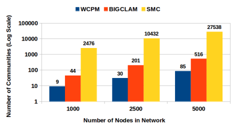
[6]Implementation available at http://snap.stanford.edu/snap/



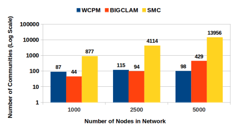
(a) Bibsonomy



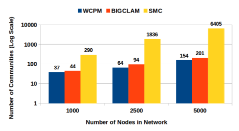
(b) Flickr



(c) IMDB

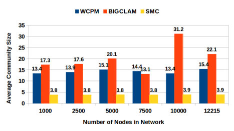


(d) Medline

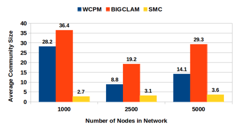


(e) Wikipedia

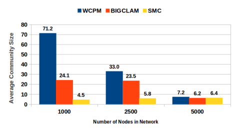
Figure 4.2 The number of communities discovered by different methods (Weighted CPM [23], BIGCLAM [107] and SMC) over networks of different sizes across all datasets. (WCPM - Red, BIGCLAM - Blue, SMC - Yellow)



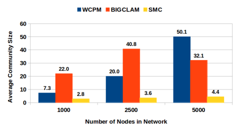
(a) Bibsonomy



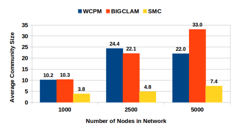
(b) Flickr



(c) IMDB



(d) Medline



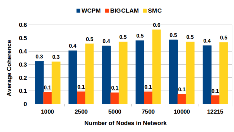
(e) Wikipedia

Figure 4.3 Average size of communities discovered by different methods (Weighted CPM [23], BIGCLAM [107] and SMC) over networks of different sizes across all datasets. (WCPM - Red, BIGCLAM - Blue, SMC - Yellow)

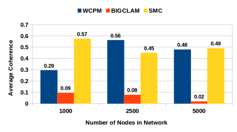
We first understand the structural properties of the communities discovered by all these algorithms and show the effectiveness of SMC over the other two methods in terms of both modularity [66] and community-based entity recommendation task.

4.6.1 Structural Properties of the Communities Discovered

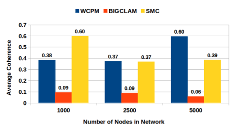
We study the structural properties of the communities discovered by SMC along with the communities discovered by WCPM [23] and BIGCLAM [107]. The entity networks were generated using the procedure described in Chapter 3, with a $\theta_{consy} = 0.001$ for noise removal step. To do a fine-grained analysis, we build entity networks of different sizes (in terms of number of nodes/entities), find communities over these networks and study their properties. For bibsonomy, we build entity networks with 1000, 2500, 5000, 7500, 10000 and 12215 entities, whereas for all other datasets, we use entity networks of 1000, 2500 and 5000 entities, due to computational constraints. We compare the structure



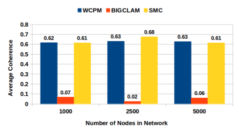
(a) Bibsonomy



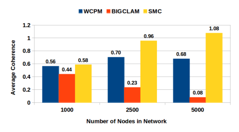
(b) Flickr



(c) IMDB

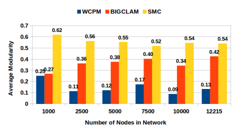


(d) Medline

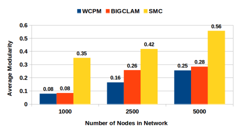


(e) Wikipedia

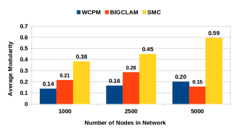
Figure 4.4 Average Coherence of communities discovered by different methods (Weighted CPM [23], BIGCLAM [107] and SMC) over networks of different sizes across all datasets. (WCPM - Red, BIGCLAM - Blue, SMC - Yellow)



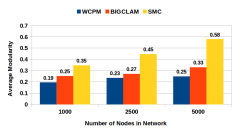
(a) Bibsonomy



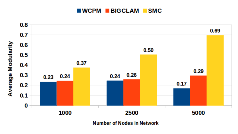
(b) Flickr



(c) IMDB



(d) Medline



(e) Wikipedia

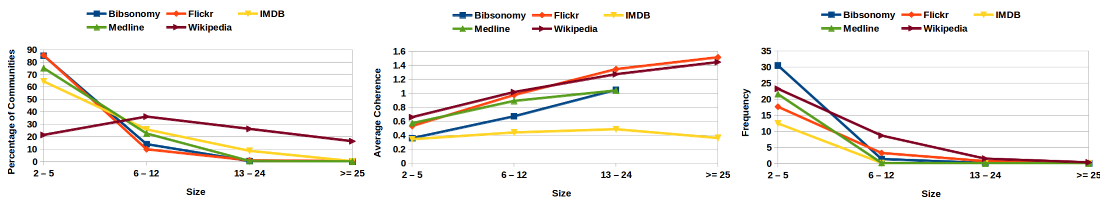
Figure 4.5 Modularity [66] of communities discovered by different methods (Weighted CPM [23], BIGCLAM [107] and SMC) over networks of different sizes across all datasets. (WCPM - Red, BIGCLAM - Blue, SMC - Yellow)

of communities discovered by the different methods in terms of their number, their average size, their average coherence and their modularity.

Figures 4.2 and 4.3 shows the number of communities discovered by the different methods and their average size for entity networks of different sizes over all datasets. It can be seen that SMC discovers large number of communities, each of small size, in comparison to WCPM and BIGCLAM. The number of SMC communities are in multiples of thousands, whereas WCPM and BIGCLAM communities are in multiples of tens and hundreds. On the contrary, the average size of SMC communities are in the range of 3-10 whereas those of WCPM and BIGCLAM are atleast 5-10 times bigger than SMC communities. Unlike other methods, SMC, being a greedy approach, aims to find coherent communities in a restricted search space and hence its communities are small in size. The number of communities discovered by SMC can be modulated by choosing the appropriate seeds, to be given as input for the grow-shrink algorithm. Here, we have used all possible edges (entity-pairs) in the network as seeds for discovering communities. Duplicate communities were removed for all the experiments.

Next, we look at two important properties, which help us understand the structure of the communities discovered. First one is coherence, which captures the quality of a community in terms of importance scores of the entities constituting the community. The SMC algorithm is based on optimizing coherence as its objective function. Second one is well-known modularity [66], which captures the structural information within the community relative to a null model of a random graph with the same degree distribution. Many standard community detection algorithms [68, 19, 62, 21, 69] are based on optimizing modularity.

Figure 4.4 shows the average coherence of the communities discovered by different methods over all datasets. In general, SMC communities were found to have high coherence (> 0.5) as it their objective function. Surprisingly, WCPM communities were also found to have coherence comparable to SMC communities. Like in SMC, the fundamental notion of cliqueness used in WCPM can be attributed to high coherence scores of communities discovered by WCPM. But the percolation factor (k) of WCPM loosens up the tightness of the communities being discovered, leading to small number of coherent communities of large size, which causes serious drawbacks of using WCPM communities in applications like entity recommendation (Section 4.6.2). Coherence of BIGCLAM communities was found to be very low. One possible reason for this could be that BIGCLAM was built for unweighted networks, unlike WCPM and SMC which used the weight information for finding coherent communities.



(a) Community Distribution (b) Average Coherence Distribution (c) Frequency Distribution

Figure 4.6 Figure shows the distribution of SMC communities, their average coherence and their frequency (in entity-set data) by size over all datasets. As the size of communities increase the coherency of the communities increase, even though the number of such large communities and their presence in the dataset decreases.

We show the modularity of communities discovered in Figure 4.5. SMC communities were found to have high modularity than communities of WCPM and BIGCLAM over all datasets. In Bibsonomy, BIGCLAM communities had significantly higher modularity than WCPM communities, whereas over other datasets, their modularity scores were comparable (slightly in favour of BIGCLAM). On average, communities of SMC were found to have modularity greater than > 0.5, which represents good partition of network. However, modularity is strictly and exclusively dependent on the graph structure and, as pointed in [34], the peak value of modularity does not in general coincide with the correct or best communities.

Another important structural property of SMC communities discovered is shown in Figure 4.6. Except in wikipedia, more than 90% of SMC communities are in the size range of 2-5. In wikipedia, the distribution is more even. Interestingly, as the communities size increase the coherency of the communities also increase, but the number of such communities and their presence in the entity-sets decrease. Hence, using frequency as a criteria for finding coherent communities [2] would not be a correct choice.

4.6.2 Community-based Entity Recommendation

In Table 4.4, we illustrate the community based entity prediction performances of all methods across all entity networks. Entity networks with 5000 nodes and θ_{consy} = 0.001 were used for experiments, except in Bibsonomy where entity network with 12215 nodes were used. Results of WCPM reported here correspond to the best value of k. Communities generated by SMC significantly outperform those obtained by WCPM and BIGCLAM in all aspects of the evaluation. We make the following detailed observations on the empirical evaluations.

Number, average size and modularity of Communities: We could see that the SMC produces a large number of communities compared to WCPM and BIGCLAM. In terms of size, SMC produces communities of relatively small size compared to WCPM and BIGCLAM. In Flickr, IMDB and Medline, SMC produces more than 10,000 communities, whereas other methods produce less than 1000. Also, average size of SMC communities was about 5 compared to others which had significantly larger

	Bibsonomy			Flickr			IMDB		
	WCPM	BC	SMC	WCPM	BC	SMC	WCPM	BC	SMC
N	113	429	7359	452	917	17023	85	516	27538
S	15.37	22.06	4.69	14.13	29.26	3.56	7.21	6.21	6.39
M	0.13	0.42	**0.54**	0.25	0.28	**0.55**	0.20	0.15	**0.59**
P(%)	1.27	2.43	**18.04**	0.33	0.26	**13.28**	1.07	1.53	**11.29**
R(%)	0.78	1.84	**14.48**	0.30	0.25	**11.96**	0.13	0.64	**10.07**
F(%)	0.97	2.09	**16.07**	0.31	0.26	**12.59**	0.24	0.90	**10.64**
P@1(%)	0.34	1.37	**16.84**	0.15	0.13	**12.37**	0.12	0.12	**13.22**
P@5(%)	1.66	3.49	**23.27**	0.29	0.29	**16.75**	1.73	1.38	**15.04**

	Medline			Wikipedia		
	WCPM	BC	SMC	WCPM	BC	SMC
N	98	429	13956	154	201	6405
S	50.07	32.1	4.37	22.01	33	7.36
M	0.25	0.33	**0.58**	0.17	0.29	**0.69**
P	0.19	0.17	**8.76**	1.24	0.83	**25.66**
R	0.08	0.07	**3.20**	0.64	0.52	**6.04**
F	0.11	0.10	**4.69**	0.84	0.64	**9.78**
P@1	0.06	0.08	**5.96**	0.24	0.33	**10.72**
P@5	0.21	0.17	**9.10**	0.41	0.62	**31.32**

Table 4.4 IR performance of WCPM [23], BC (BIGCLAM) [107] and SMC-based communities in entity prediction. N, S, M, P, R, F denotes No. of Communities, Average size of Communities, Modularity [66] of the communities, Precision, Recall and F-measure respectively and P@1, P@5 denotes precision at one and five predictions respectively. Networks with 5000 nodes and θ_{consy} = 0.001 were used for experiments, except in Bibsonomy where network with 12215 nodes were used. Results of WCPM reported here correspond to the best value of k.

average size (> 15). Also, in terms of modularity, SMC was found be produce highly modular communities (> 0.5) compared to other methods. A complete discussion on this can be found in Section 4.6.1.

Precision, Recall and F-measure: SMC performs exceptionally better than other methods over all entity networks. Over all datasets, the average precision of SMC communities is 15.41% compared to WCPM and BIGCLAM which have 0.82% and 1.04% respectively. Similarly, the average recall of SMC is 9.15% compared to 0.39% and 0.684% of WCPM and BIGCLAM communities. As a result, the average F-measure of SMC, BIGCLAM and WCPM communities are 10.74%, 0.80% and 0.49% respectively, which is a stupendous improvement compared to other methods in a application-oriented task of entity recommendation. The first reason for such improvement is the restricted search space of the grow-shrink algorithm along with the highly conservative notion of coherence as the objective function which lead to the discovery of meaningful and strong entity concepts. The second reason is the degree of exhaustiveness of the method. SMC by design is completely exhaustive in the way it uses the seeds. The overall precision and recall is low because communities by themselves are not enough for recommending entities in a recommendation system. We use the recommendation system only as an evaluation metric for comparing methods. Significant improvements could also be seen on the Precision@1 and Precision@5 scores also, which are very important metrics in the task of recommendation.

Overall from the quality of communities as well as the time taken to find the communities, SMC is shown to be a competitive alternative. The new cohesiveness measure and the notion of local node centrality are comparable or sometimes better measures of community-ness compared to modularity [66] or other measures.

4.6.3 Examples of SMC Communities

To illustrate the quality of the communities, we show some examples of some communities discovered by SMC over all datasets in Table 4.5. We assigned context name to each community for ease of browsing the results. In general, these communities are quite compact, precise, and meaningful. Due to the highly conservative definitions of the coherence (minimum of importance) and Soft Maximal Cliques (local maxima of coherence), these communities turn out to be highly compact and addition of even one entity might bring the coherence down.

4.7 Comparison with LDA

Latent Dirichlet Allocation (LDA) [79] is a very prominent topic-modelling method with applications to document classification/clustering, document summarization, and text/web mining community in general. The communities detected by SMC can also be described as semantic concepts discovered from entity-set data. We make a qualitative comparison of the communities of SMC and topics of LDA on the task of entity recommendation. We use the same experimental setup of entity recommendation

(a) Bibsonomy

Context	Entity Communities
Medical Journals	journals, medicine, medical, reviews, peerreviewed, studies, biomedical, physicians.
Operating System	linux, debian, ubuntu, unix, opensource, os, software, freeware, microsoft, windows, mac, computer.
Recommenders	personalization, recommender, collaborative, tagging, folksonomy, recommendersystems, ecommerce, filtering, usermodels, clustering.

(b) IMDB

Context	Entity Communities
Photo-shoot	appearance, elimination, criticism, makeover, style, challenge, modeling, clothing, fashion, magazine, makeup, photo-shoot, winner, model, photography.
Film	movie-studio, movie-set, movie-producer, movie-director, film-making, hollywood, actor, actress, movie-star, hollywood-california.
Family relationships	father-daughter-relationship, mother-daughter-relationship, brother-sister-relationship, sister-sister-relationship, family-relationships

(c) Flickr

Context	Entity Communities
Fire-Department	firefighter, firemen, fireman, firetruck, fire, engine, smoke, feuer- wehr, firefighters, department, flames.
Airplanes	airplane, aircraft, plane, aviation, airbus, boeing, aeroplane, flight, airlines.
Sketch	drawing, pencil, sketch, ink, illustration, draw, pen.

(d) Medline

Context	Entity Communities
Mycoses	Aspergillosis, Antifungal Agents, Amphotericin B, Candidiasis, Immunocompromised Host, Lung Diseases Fungal, Mycoses, Dermatomycoses.
Bone Marrow	Bone Marrow Cells, Bone Marrow, Hematopoietic Stem Cells, Hematopoiesis, Colony-Forming Units Assay.
Peptic Ulcer	Helicobacter pylori, Helicobacter Infections, Gastritis, Dyspepsia, Peptic Ulcer, Duodenal Ulcer, Gastroscopy.

(e) Wikipedia

Context	Entity Communities
Automobiles	Automotive industry, Internal combustion engine, Car body style, Car classification, Transmission (mechanics), Wheelbase, Automobile, Automobile layout, Sedan (car), Sports car, Ford Motor Company, Straight-4, Toyota, BMW, Chevrolet, Porsche, Mercedes-Benz, Honda.
Ivy League	Yale University, Harvard University, Columbia University, Princeton University, Dartmouth College, Cornell University, Brown University.
IT	Amazon.com, Yahoo!, EBay, Google, Sony, Apple Inc., Sun Microsystems, Microsoft, Information technology, Electronic Arts, Sega, Nintendo.

Table 4.5 Examples of Communities discovered by SMC

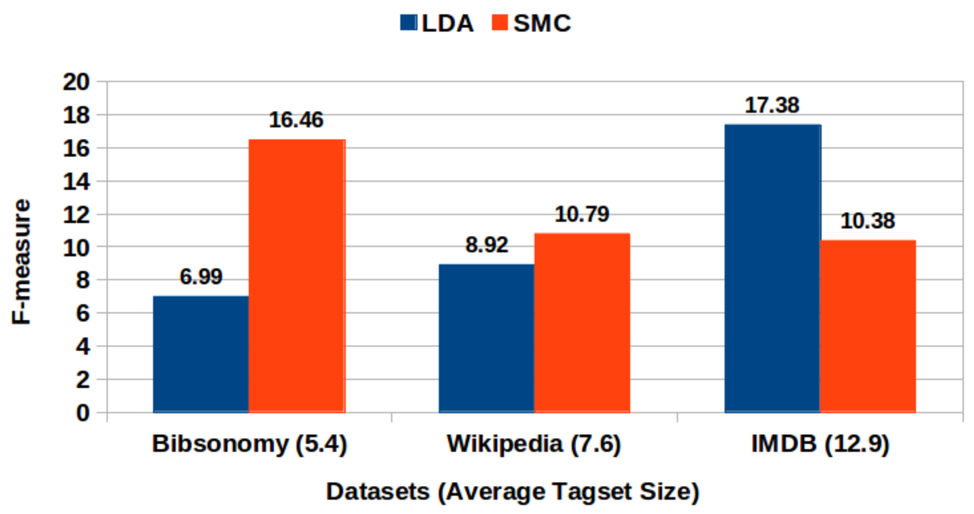


Figure 4.7 Comparison of F-measures of SMC communities with LDA [79] topics in the task of entity recommendation over Bibsonomy, IMDB and Wikipedia datasets of size 1000. The number of topics is kept same as the number of communities discovered by SMC. The average entity-set size of each dataset is provided in brackets.

for LDA also. Considering every entity-set as a document of entity [7] and the vocabulary of entities as the vocabulary of terms, we apply LDA of training documents and learns topics from them. For a fair comparison, the number of topics is kept same as the number of communities discovered by SMC. Due to computation requirements of LDA, we perform this experiment on Bibsonomy, IMDB and Wikipedia networks of size 1000. Figure 4.7 compares the F-measures of the SMC communities and LDA topics in the task of entity recommendation. We could see that as performance of LDA is low for datasets of low entity-set size compared to LDA. In Bibsonomy where the average entity-set size is 5.4, there is 135.47% improvement in F-measure of SMC compared to LDA, whereas in IMDB where the average entity-set is 12.9, there is 67.44% decrease in F-measure of SMC compared to LDA. Hence, LDA would not be the right choice for semantic concept modelling in tagging systems, where the average length of entity-set (document) is very low and the entity (term) frequencies in entity-sets (documents) is either zero or one.

4.8 Effect of Consistency Threshold

Consistency score defines the magnitude of associativity between entities in entity-set data. Here, we study the effect of consistency threshold in the formation of communities, in terms of F-measure with respect to task of entity recommendation. As the threshold is increased, the number of edges in the co-occurrence consistency graph would decrease, resulting in drop in density of graph as well as decrease in size and number of communities discovered from the graph. As a result, the precision-recall

[7]Every term in the document has a frequency of one, unlike in usual scenarios.

θ_{consy}	N	S	P(%)	R(%)	F(%)
0.001	7053	3.89	17.15	13.61	15.18
0.005	7029	3.87	17.02	13.52	15.07
0.01	6969	3.84	17.1	13.57	15.13
0.05	6224	3.6	16.39	12.98	14.49
0.1	5088	3.37	15.03	11.79	13.21
0.2	3039	3.21	13.01	8.57	10.34
0.25	2185	3.30	14.61	7.41	9.84

Table 4.6 Comparison of SMC communities entity recommendation performance on Bibsonomy Dataset (Network Size = 12215) over different values of consistency threshold (θ_{consy}). N, S, P, R, F denotes No. of Communities, Average size of Communities, Precision, Recall and F-measure respectively

in the entity recommendation task also decreases, resulting in low F-measure scores. Table 4.6 supports our claim.

Chapter 5

Compacting Large and Loose Communities

Detecting compact overlapping communities in large networks is an important pattern recognition problem with applications in many domains. Most community detection algorithms trade-off between community sizes, their compactness and the scalability of finding communities. Clique Percolation Method (CPM) [76] and Local Fitness Maximization (LFM) [47] are two prominent and commonly used overlapping community detection methods that scale with large networks. However, significant number of communities found by them are large, noisy, and loose. In this chapter, we propose a general algorithm that takes such large and loose communities generated by any method and refines them into compact communities in a systematic fashion. We use a new measure of community-ness based on eigenvector centrality [75], identify loose communities using this measure and propose an algorithm for partitioning such loose communities into compact communities. We refine the communities found by CPM and LFM using our method and show their effectiveness compared to the original communities in a recommendation engine task.

5.1 Introduction

Unsupervised pattern recognition tasks such as clustering, density estimation, outlier detection, dimensionality reduction, etc. are used to understand the underlying nature of the data, find latent structures within the data, and derive useful features from it. One such important unsupervised learning task on network or graph data is to find compact overlapping communities i.e. groups of nodes in the graph that are tightly connected to each other. This has applications in many domains such as Biology, Social Networking [33], Web Mining [26], etc. Also, communities in networks often overlap as nodes can belong to multiple communities at once. For example, researchers might belong to more than one research community.

Most community detection algorithms strive to strike a balance between community *size* and their *compactness*. Oversized communities might contain unnecessary noise while undersized communities might not generalize the concept well enough. Another trade-off in community detection is that of *compactness* and *scalability*. Finding large number of compact communities such as maximal cliques is an NP-hard problem [82], making them impractical for large networks. Because of these trade-offs, existing community detection algorithms find several communities that are large, noisy, and loose, which pose significant problems in using them in many applications of communities like recommendation systems [83], semantic retrieval, semantic user profiling, conceptual browsing [30] etc.

In this chapter, we address this problem of compacting and cleaning such large and loose communities generated by any method into small and compact communities. We use the novel and natural measure of community-ness called coherence, defined in section 3.2.2, to determine whether a community is compact and if not, it is greedily partitioned into smaller communities until each of the sub-communities are compact. Our greedy algorithm, called hereafter the **Loose Community Partition** (LCP) algorithm, iterates over two phrases: (i) *Shrink Phase* that removes the most noisy nodes in the current community and generates a compact candidate seed community, and (ii) *Grow Phase* that enriches this candidate seed community by adding the most related nodes (if any) to it. Note that LCP does not partition every large-size community, but only loose communities as explained in section 5.5.

Extensive evaluations on large real world datasets like Amazon [106] and Flickr [55], show that the proposed algorithm significantly cleans up these large noisy communities into compact and high precision communities. We robustly evaluate LCP using an unsupervised metric that measures the "average overlapping community modularity" [71]. We also show application of our method in real world, by building a community based product/tag recommendation system and measure the precision and recall as our supervised metric for evaluation.

The rest of the chapter proceeds as follows. In next section, we describe two popular and state-of-art community detection methods in detail. Section 5.3 illustrates the problem with these community detection methods with illustrations. In section 5.5 we introduce our LCP algorithm for compacting communities and explain it in detail. The experimental evaluations are presented in section 5.6.

5.2 Popular Community Detection Methods

Two popular and commonly used overlapping community detection algorithms are the Clique Percolation Method (CPM) and the Local Fitness Maximization (LFM). CPM by Palla $et\ al.$ [76], is based on the belief that communities are unions of adjacent k-cliques (complete graphs with k nodes) and that inter-community regions of the network do not possess such strong edge density. LFM [47] is a well known greedy algorithm that grows a seed node into a community by maximizing the modularity [66] of the community. In the following subsections, we describe these two popular community detection methods in detail.

5.2.1 Clique Percolation Method

The Clique Percolation Method [76] is a popular approach for analyzing the overlapping community structure of networks. It builds up the communities from k-cliques, which correspond to complete (fully

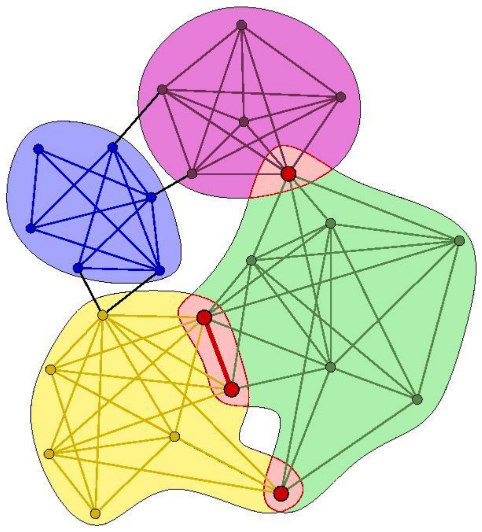


Figure 5.1 Illustration of k-clique communities at $k = 4$. Communities are color-coded and the overlap between them is emphasized in red.

connected) sub-graphs of k nodes. For example, a k-clique at $k = 3$ is equivalent to a triangle. Two k-cliques are considered adjacent if they share k-1 nodes. A community is defined as the maximum union of k-cliques that can be reached from each other through a series of adjacent k-cliques. Such communities can be best represented with the help of a k-clique template (an object isomorphic to a complete graph of k nodes). Such a template can be placed onto any k-clique in the graph, and rolled to an adjacent k-clique by relocating one of its nodes and keeping its other k-1 nodes fixed. Thus, the k-clique communities of a network are all those sub-graphs that can be fully explored by rolling a k-clique template in them, but cannot be left by this template.

This definition allows overlaps between the communities in a natural way, as illustrated in Figure 5.1, showing four k-clique communities at $k = 4$. The communities are color-coded and the overlap between them is emphasized in red. The definition above is also local: if a certain sub-graph fulfills the criteria to be considered as a community, then it will remain a community independent of what happens to another part of the network far away. In contrast, when searching for the communities by optimizing with respect to a global quantity, a change far away the network can reshape the communities in the unperturbed regions as well. Furthermore, it has been shown that global methods suffer from a resolution limit problem [28], where the smallest community that can be extracted is dependant on the system size. A local community definition such as here circumvents the problem automatically.

Since even small networks can contain a vast number of k-cliques, the implementation of this approach is based on locating the maximal cliques rather than the individual k-cliques. Thus, the complexity of this approach in practice is equivalent to that of the NP-hard maximal clique finding [12]. This means that although networks with few million nodes have already been analyzed successfully with this approach, no prior estimate can be given for the runtime of the algorithm based simply on the system size.

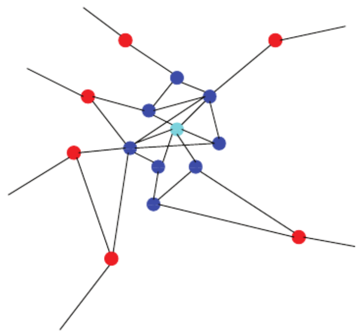


Figure 5.2 Schematic example of natural community for a node (sky-blue point in the figure) according to our definition. The blue nodes are the other members of the group and have positive fitness within the group, while the red nodes have all negative fitness with respect to the group.

5.2.2 Local Fitness Maximization

The Local Fitness Maximization method [47] performs a local exploration of the network, searching for the natural community of each node. The basic assumption is that communities are essentially local structures, involving the nodes belonging to the modules themselves plus atmost an extended neighborhood of them. Here, a community is a subgraph identified by the maximization of a property or *fitness* of its nodes. The fitness used here, is defined as:

$$f_G = \frac{k_{in}^G}{(k_{in}^G + k_{out}^G)^\alpha} \tag{5.1}$$

where k_{in}^G and k_{out}^G are the total internal and external degrees of the nodes of module G, and α is a positive real-values parameter, controlling the size of the communities. The internal degree of a module equals double the number of internal links of the module. The external degree is the number of links joining each member of the module with the rest of the graph. The aim of this work is to determine a subgraph starting from a node, say A, such that inclusion of a new node, or elimination of one node from the subgraph would lower f_G. They call such a subgraph the *natural community* of node A. This amounts to determining local maxima for the fitness function for a given α.

Given the fitness function f_G, the fitness of a node A with respect to a subgraph G, f_G^A, is defined as the variation of the fitness of the subgraph G with and without node A, i.e.

$$f_G^A = f_{G+\{A\}} - f_{G-\{A\}} \tag{5.2}$$

In equation (5.2), the symbol $G + \{A\}$ ($G - \{A\}$) indicates the subgraph obtained from module G with node A inside (outside). The natural community of node A is identified with the following procedure. Let us suppose that we have covered a subgraph G including node A. Initially, G is identified with node A ($k_{in}^G = 0$). Each iteration of the algorithm consists of the following steps:

1. a loop is performed over all neighboring nodes of G not included in G.

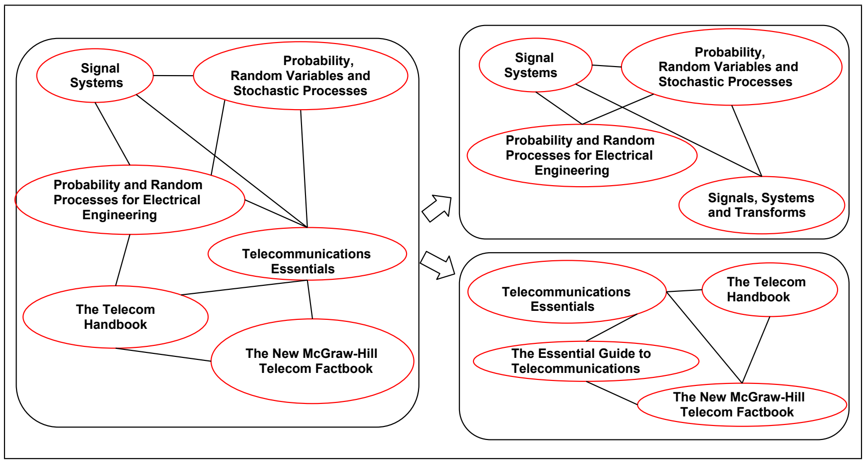

Signal Systems
Probability, Random Variables and Stochastic Processes
Probability and Random Processes for Electrical Engineering
Telecommunications Essentials
The Telecom Handbook
The New McGraw-Hill Telecom Factbook
Signals, Systems and Transforms
The Essential Guide to Telecommunications

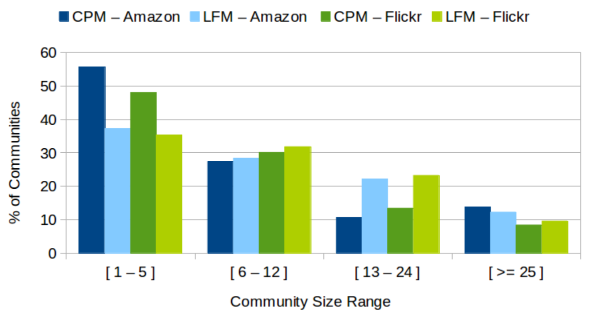
Figure 5.3 Figure shows example of our LCP algorithm on a CPM community of books, in Amazon dataset. The CPM community contains sub-communities of books, of two related electrical engineering topics: (i) One on Signal Processing and, (ii) Other on Telecommunications. The Shrink Phase of our LCP algorithm partitions the CPM community into two compact sub-communities. The Grow Phase enriches the sub-communities by adding related books to the sub-communities, like *Signal, Systems and Transforms* and *The Essential Guide to Telecommunications*, respectively.

2. the neighbor with the largest fitness is added to G, yielding a larger subgraph G'.
3. the fitness of each node of G' is recalculated.
4. if a node turns out to have negative fitness, it is removed from G', yielding a new subgraph G''.
5. if step 4 occurs, repeat from step 3, otherwise repeat from step 1 for subgraph G''.

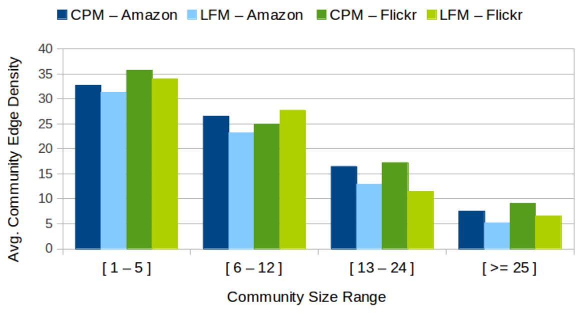
The process stops when the nodes examined in step 1 all have negative fitness (Figure 5.2). This procedure corresponds to a sort of greedy optimization of the fitness function, as at each move one looks for the highest possible increase.

5.3 Problem with Existing Community Detection Methods

A problem with communities discovered using CPM [76] and LFM [47] is that a significant number of them are large in size and loosely associated. Figure 5.3 (left part of the figure) shows example of a community of books discovered by CPM in Amazon dataset, which is reasonably large and loose as it contains sub-communities of two highly related electrical engineering topics. To appreciate the seriousness of the problem, we show the community frequency distribution across various community size buckets for two datasets, Amazon and Flickr (Figure 5.4(a)). On average, 23.17% of CPM communities and 33.66% of LFM communities are of size $>= 13$ in both datasets. This shows the scope and significance of the large community size problem. Figure 5.4(b) shows the average community density across various community size buckets for the two datasets. Edge density of a community typically is a good



(a) Community Frequency Distribution by size



(b) Average community density by size

Figure 5.4 Community frequency distribution and average community density based on size using CPM [76] ($k = 3$) and LFM [47] ($\alpha = 1$) on Amazon and Flickr data. Significant number of communities ($> 25\%$), which are typically large in size ($>= 13$), have very low edge density($\approx 10\%$).

metric for measuring the structure within a community. Low edge density scores indicate loose structure within the community. As expected, there is a strong inverse correlation between community size and their densities. The point to note here is that the average density of the large size communities ($>= 13$) is 12.59% for CPM and 9.04% for LFM, which is almost three times less than the average densities of smaller communities, illustrating the looseness in the community structure of large communities. This is primarily because, in real-world networks, there is a lot of density variations in different regions of the graph and existing community detection methods have no mechanism to adapt their parameters to different regions of the network based on the network densities. Note that not every large-size community is loose, only experimentally (Figure 5.4) it has been observed that more often than not large-sized communities are typically loose.

5.4 Compactness and Neighborhood of a Community

Before we introduce the LCP algorithm, we first need to define the compactness of a community. Most of the existing community-ness measures [74, 66] involve optimization of some local objective function and the communities discovered by optimizing them are usually loose and large in size. We use the concept of eigenvector centrality [75] to define the compactness of a community. In chapter 3 (sections 3.2.1 and 3.2.2), we introduced the notions of local node centrality and coherence to define the importance of a node within the community and community-ness of the community respectively. We make use of these notations and define the compactness of a community as its the coherence. On similar terms, we use the neighborhood definition of a community as described in section 3.2.3, in terms of up-neighbors and down-neighbors, to explain the LCP algorithm for partitioning loose communities.

5.5 LCP Algorithm for Partitioning Loose Communities

As we have defined our notions of coherence and the neighborhood of a community, here we discuss our LCP algorithm for partitioning loose communities into compact communities. We start by defining two important operations of our greedy algorithm: (i) grow operation and (ii) shrink operation. The grow operation finds the highest coherence up-neighbor of **x** in $\mathcal{N}_+(\mathbf{x})$, by finding the best node (and relevant edges) from $\mathbf{V}-\mathbf{x}$, to add to **x**. The shrink operation finds the highest coherence down-neighbor of **x** in $\mathcal{N}_-(\mathbf{x})$. Detailed discussions of the grow and shrink operations along with illustrations can be found in section 3.2.4.

Our algorithm for partitioning communities is iterative and involves three major phases:

- **Shrink Phase**, where we iteratively apply the shrink operation on the input community, until the coherence of the community keeps increasing. Each shrink operation will result in removal of least important node from the community. As output, we will have (i) a set of nodes left in the input community (candidate set) and, (ii) a set of nodes removed during the shrink operations on the input community (residue set).

- **Grow Phase**, where we iteratively apply the grow operation on the candidate set, until the coherence of the candidate set keeps increasing. Each grow operation will result in addition of a correlated node (if any) to the candidate set. The output of this phase would be a strong and compact community.

- **Final Phase**, where we send the residue set as input to shrink phase, until there is no residue set left or no further shrink is possible.

Given a loose community $\mathbf{x}_0$ and a network Φ, our greedy iterative method for partitioning loose communities is shown in Algorithm 3. Figure 5.3 shows an example of partition of a loose community into two compact sub-communities by the shrink phase and the further enhancement of the sub-communities by the grow phase. Given the strong intuition behind the coherence measure and the dependency of our algorithm on coherence at each step, the possibility of partition of strong, clean communities reduces by large extent, even if the community is of large size.

5.6 Experimental Evaluation

Here, we compare the communities obtained using CPM and LFM, with communities obtained after applying the LCP algorithm on the CPM and LFM communities (LCP-CPM and LCP-LFM), on different metrics over two real world datasets.

Algorithm 3 Loose Community Partition ($\mathbf{x}_0$, Φ)

```
1: $\mathbf{x} \leftarrow \mathbf{x}_0$
2: $\mathbf{x}_{compact}$ = [ ]
3: loop
4:     [$\mathbf{x}_{candidate}, \mathbf{x}_{residue}$] = ShrinkPhase($\mathbf{x}$, $\Phi$)
5:     $\mathbf{x}_{compact}$ = [ $\mathbf{x}_{compact}$ GrowPhase($\mathbf{x}_{candidate}$, $\Phi$) ]
6:     $\mathbf{x} \leftarrow \mathbf{x}_{residue}$
7: end loop
8: return $\mathbf{x}_{compact}$
9: A. ShrinkPhase($\mathbf{x}$, $\Phi$)
10: loop
11:     $\mathbf{x}_{residue}$ = [ ]
12:     $[\mathbf{x}^-, \mathbf{x}_{removed}] \leftarrow Shrink(\mathbf{x}, \Phi)$ {Best down-neighbor.}
13:     if $\pi(\mathbf{x}^-) > \pi(\mathbf{x})$ then
14:         $\mathbf{x} \leftarrow \mathbf{x}^-$ {Not reached maximum coherence yet.}
15:         $\mathbf{x}_{residue}$ = [ $\mathbf{x}_{residue}$ $\mathbf{x}_{removed}$ ]
16:     else
17:         return [$\mathbf{x}$, $\mathbf{x}_{residue}$] {maximum coherence.}
18:     end if
19: end loop
20: B. GrowPhase($\mathbf{x}$, $\Phi$)
21: $\mathbf{x}^+ \leftarrow Grow(\mathbf{x}, \Phi)$ {Best up-neighbor.}
22: while $\pi(\mathbf{x}^+) > \pi(\mathbf{x})$ do
23:     $\mathbf{x} \leftarrow \mathbf{x}^+$
24:     $\mathbf{x}^+ \leftarrow Grow(\mathbf{x}, \Phi)$
25: end while
26: return $\mathbf{x}$
```

Dataset	Nodes	Edges
Amazon	548,552	925,872
Flickr	5,000	30,006

Table 5.1 Statistical properties of Amazon and Flickr datasets

Datasets	CPM	LCP-CPM	LFM	LCP-LFM
Amazon	0.27	**0.33**	0.38	**0.48**
Flickr	0.31	**0.37**	0.43	**0.52**

Table 5.2 Overlapping modularity scores of the communities discovered by CPM [76] ($k = 3$), LFM [47] ($\alpha = 1$), our LCP-CPM and LCP-LFM in Amazon and Flickr network.

5.6.1 Datasets & Evaluation Metrics

We use the unweighted Amazon product network and the weighted Flickr tag network for finding communities. The Amazon product co-purchasing network[1] [106] is obtained by crawling Amazon website and contains product metadata and review information of about 548,552 different products. Ground-truth communities are available for this network. The Flickr tag network[2] [54] is created using a random subset of 800,000 images from a collection of 3.5 million social-tagged images from Flickr. The weighted tag network is created by computing the statistically significant co-occurrences among tags. Since, ground truth communities are not available for this network, we divide the data into training and testing tagsets. The training tagsets are used to derive the tag network and communities are detected on this weighted tag network. The testing tagsets are used in the recommendation task. Table 5.1 shows statistical properties like the number of nodes, edges in the graph for both Amazon and Flickr networks.

We use two different types of methods for evaluations: overlapping modularity - a standard, unsupervised metric and product/tag recommendation - an application-oriented supervised metric for evaluating communities, described in detail in section 4.2.

5.6.2 Comparison of LCP communities with CPM and LFM communities

Table 5.2 shows the overlapping modularity [71] of the communities discovered by CPM, LFM and our LCP-CPM, LCP-LFM on Amazon and Flickr networks. In the Amazon network, LCP leads to 22.2% increase in modularity over CPM communities and 26.31% increase over LFM communities. Similarly, in Flickr network, there is an 19.35% increase in modularity over CPM communities and 20.92% increase over LFM communities. These significant improvements are due to the use of LCP algorithm. Also note that LFM is primarily a modularity maximization method and even on that, LCP leads to a significant improvement in the modularity metric.

[1]http://snap.stanford.edu/data/com-Amazon.html

[2]http://staff.science.uva.nl/ xirong/index.php?n=DataSet.Flickr3m

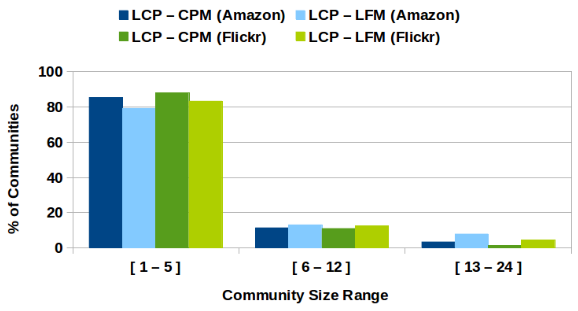
(a) Amazon

	CPM	LCP-CPM	LFM	LCP-LFM
N	28,402	38,040	10,318	18,482
S	10.36	6.54	14.27	6.17
P(%)	34.13	**38.74**	24.36	**27.58**
R(%)	8.97	**10.91**	6.12	**8.83**
F(%)	14.21	**17.02**	9.78	**13.37**
P@1(%)	19.54	**20.23**	12.58	**15.19**
P@5(%)	37.91	**47.07**	25.39	**38.21**

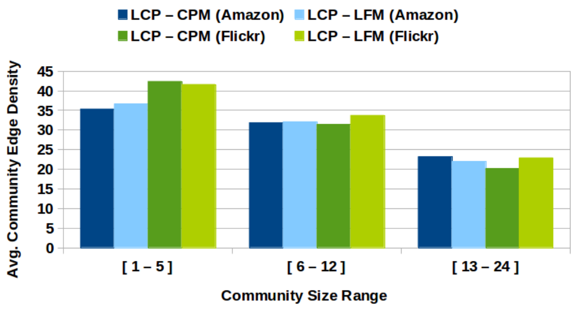
(b) Flickr

	CPM	LCP-CPM	LFM	LCP-LFM
N	1138	1342	712	1023
S	11.81	5.19	13.87	6.37
P(%)	17.72	**22.29**	13.53	**18.12**
R(%)	15.48	**18.98**	9.54	**13.13**
F(%)	16.53	**20.5**	11.19	**15.22**
P@1(%)	13.59	**24.61**	7.21	**15.45**
P@5(%)	22.70	**36.1**	17.17	**30.51**

Table 5.3 Performance of CPM [76](k = 3), LFM [47](α = 1), LCP-CPM and LCP-LFM communities in product/tag recommendation. N, S, P, R, F denotes Number of Communities, Avg. size of Community, Precision, Recall and F-measure respectively and P@1, P@5 denotes precision at one and five predictions respectively.



(a) Community Frequency Distribution by size



(b) Average community density by size

Figure 5.5 Community frequency distribution and the average community edge density, after applying LCP algorithm on the communities obtained using CPM and LFM on both Amazon and Flickr.

In Table 5.3, we compare the community based product/tag recommendation performances of CPM and LFM communities with its LCP counterparts over Amazon product and Flickr tag networks. Communities discovered using LCP significantly outperform CPM and LFM in all aspects of the evaluation. While the **average size** of LCP communities ($\approx$ 6.06) are significantly smaller than the average size of their non-LCP counterparts ($\approx$ 12.57), the **number of communities** increases significantly (25.92% for CPM, 61.39% for LFM). This is because each large, loose community is partitioned by LCP into a number of small but compact communities. Significant improvement is also seen over **precision, recall and F-measure**, in task of product/tag recommendations. Compared to CPM and LFM communities, we observe, on average, 19.65% and 23.57% increase in precision, 22.11% and 40.95% increase in recall, thus resulting in 21.89% and 36.35% increase in F-measure respectively. This partitioning of loose communities into compact ones improves their productivity, which is exemplified by their performances in the task of recommendation. Improvements could also be seen on the Precision@1 and Precision@5 scores. The overall precision and recall is low because communities by themselves are not enough for recommending products/tags in a recommendation system. We use the recommendation system only as an evaluation metric for comparing methods.

Figure 5.5 shows the community frequency distribution and average community edge density of LCP-CPM and LCP-LFM communities on Amazon and Flickr networks. Compared to the corresponding statistics of CPM and LFM-based communities, shown in Figure 5.4, we see substantial reduction in the number of large-sized communities(90.26% for CPM, 82.02% for LFM) as well as significant increase in the average community density(17.88% for CPM, 24.14% for LFM).

Hence, from compactness to recommendation performances, it can be seen that LCP improves the quality of the communities discovered by CPM and LFM by partitioning them into compact and semantically strong communities.

Chapter 6

Application to Image Annotation

Labels associated with social images are valuable source of information for tasks of image annotation, understanding and retrieval. These labels are often found to be noisy, mainly due to the collaborative tagging activities of users. Existing methods on annotation have been developed and verified on noise free labels of images. In this chapter, we propose a novel and generic framework that exploits the collective knowledge embedded in noisy label co-occurrence pairs to derive robust annotations. We compare our method with a well-known image annotation algorithm and show its superiority in terms of annotation accuracy on benchmark Corel5K and ESP datasets in presence of noisy labels.

6.1 The Task of Image Annotation

Over the years, the Internet has become the largest database for multimedia content and is organized in a rich and complex way through tagging activities. One such example is collaborative tagging websites, such as Flickr, which collects millions of photos per month from tens of thousands of users. Well-known commercial systems including Google and Yahoo rely on surrounding descriptions of images embedded in the web pages for the image retrieval. Images without clear context descriptions will either be returned as false positives or be totally discarded during the retrieval. Image auto-annotation techniques provide an attainable way to associate the "visuality" of the images with their semantics, which can be used to search unlabeled image collections, and return more relevant images to the users.

Given an input image, the goal of automatic image annotation is to assign a few relevant text keywords to the image that reflect its visual content. Utilizing image content to assign a richer, more relevant set of keywords would allow one to further exploit the fast indexing and retrieval architecture of Web image search engines for improved image search. This makes the problem of annotating images with relevant text keywords of immense practical interest. Consequently, there is immense research interest in producing efficient image annotation techniques for labelling social images to cope with the continuously growing amount of social image data.

Image annotation is a difficult task for two main reasons: First is the well-known pixel-to-predicate or semantic gap problem, which points to the fact that it is hard to extract semantically meaningful entities using just low level image features, e.g. color and texture. Doing explicit recognition of thousands of objects or classes reliably is currently an unsolved problem. The second difficulty arises due to the lack of correspondence between the keywords and image regions in the training data. For each image, one has access to keywords assigned to the entire image and it is not known which regions of the image correspond to these keywords. This makes difficult the direct learning of classifiers by assuming each keyword to be a separate class.

6.2 Prior Work

Image annotation has been a topic of on-going research for more than a decade and several interesting techniques have been proposed. Most of these treat annotation as translation from image instances to keywords. The translation paradigm is typically based on some model of image and text co-occurrences [39, 22]. The translation approach of [22] was extended to models that ascertain associations indirectly, through latent topic/aspect/context spaces [7, 65]. One such model, the Correspondence Latent Dirichlet Allocation (CorrLDA) [7], considers associations through a latent topic space in a generatively learned model. Despite its appealing structure, this class of models remains sensitive to the choice of topic model, initial parameters, prior image segmentation, and more importantly the inference and learning approximations to handle the typically intractable exact analysis.

Cross Media Relevance Models (CMRM) [41], Continuous Relevance Model (CRM) [50], and Multiple Bernoulli Relevance Model (MBRM) [24] assume different, nonparametric density representations of the joint word-image space. Alternative approaches based on graph representation of joint queries [63], and cross-language LSI [38], offer means for linking the word-image occurrences, but still do not perform as well as the non-parametric models.

Among the supervised learning approaches, K-nearest neighbour (or KNN) based methods [24, 61, 36, 97] have been found to give some of the best results despite their simplicity. The intuition is that “similar images share common labels” [61]. In most of these approaches, this similarity is determined only using image features.

6.3 Nearest Neighbour Model for Annotation

K-nearest neighbour (or KNN) based methods [61, 97, 36] have been found to give some of the best results on the task of image annotation. The intuition behind them is that similar images share common labels. Most relevant KNN-based annotation methods are (i) JEC [61] which treats image annotation as retrieval problem. Using multiple global features, a greedy algorithm is used for label transfer from neighbours. They also performed metric learning in the distance space but it could not do any better than using equal weights. This is because they used a classification-based metric learning approach for the annotation task which is multi-label classification by nature. Though JEC looks simple at the modelling level, it reported the best results on benchmark annotation datasets when it was proposed. (ii)

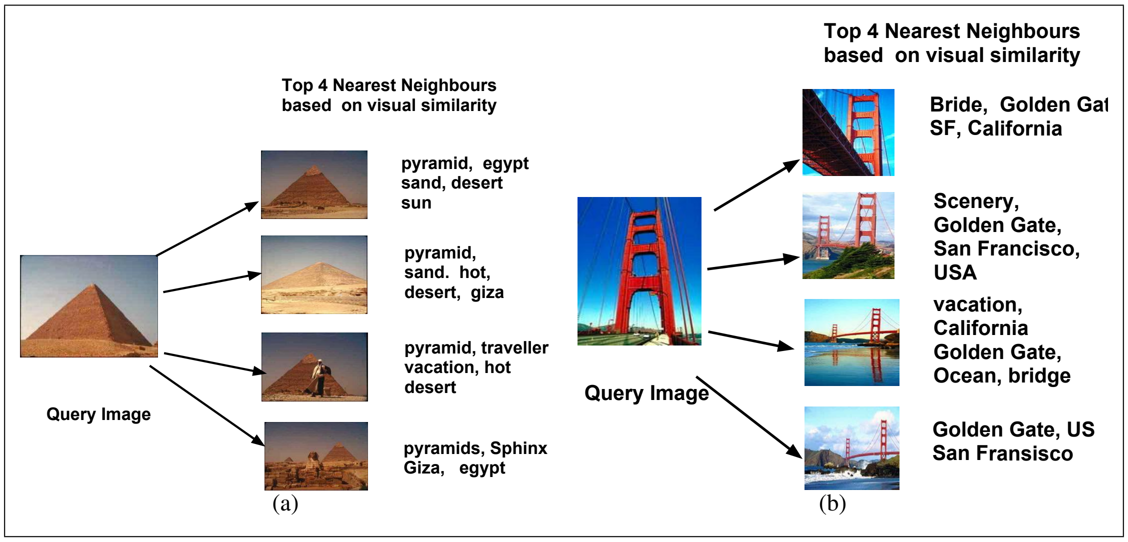

Top 4 Nearest Neighbours based on visual similarity
pyramid, egypt sand, desert sun
pyramid, sand. hot, desert, giza
pyramid, traveller vacation, hot desert
pyramids, Sphinx Giza, egypt
Query Image
(a)
Top 4 Nearest Neighbours based on visual similarity
Bride, Golden Gat SF, California
Scenery, Golden Gate, San Francisco, USA
vacation, California Golden Gate, Ocean, bridge
Golden Gate, US San Fransisco
Query Image
(b)

Figure 6.1 Figure illustrates example of the well known K-Nearest Neighbour(K-NN) model used in image annotation. For each query image, the top 4 visually similar images are shown, along with the labels associated with them. The labels of the nearest neighbours are transferred to the query image for annotation.

TagProp [36], a weighted KNN based method that transfers labels by taking weighted average of labels present among the neighbours, and (iii) 2PKNN [97], where a class-wise semantic neighbourhood is defined and only samples within this neighbourhood are used for annotation of unseen image. Since JEC is the essential backend method for modern successful techniques [97, 36], we compare our results with JEC [61] and show that our method is robust under noisy labels.

Let $I = \{ I_1, \ldots, I_N \}$ denote the collection of images and $V = \{ v_1, \ldots, v_m \}$ denote the *vocabulary* of m labels. The training set $T = \{ (I_1,V_1) \ldots (I_N,V_N) \}$ consists of pairs of images and their corresponding label sets, with each $V_i \subseteq V$. Given an unannotated image J, the task of annotation is to predict a set of labels that semantically describe J. In a typical NN-setting, we pick the top K visually similar images $T_J = \{ (T_{J,1},\gamma_{J,1}) \ldots (T_{J,K},\gamma_{J,K}) \}$. Here, $\gamma_{J,K}$ denotes the visual similarity score of image J with its K^{th} neighbour and is defined as:

$$\gamma_{J,K} = VisualSimilarity(I_J, T_{J,K}) \tag{6.1}$$

This score is generated as a function of distance between the images in visual feature space (SIFT [58], Color Histograms, GIST [73]). Then, the labels of the nearest neighbours are ranked on basis of a label scoring function, κ_{J,v_i} and the top L labels are used to annotate the test image J. This label scoring function is usually based on frequency [61] or distance [36]. Figure 6.1 shows illustration of KNN model for image annotation.

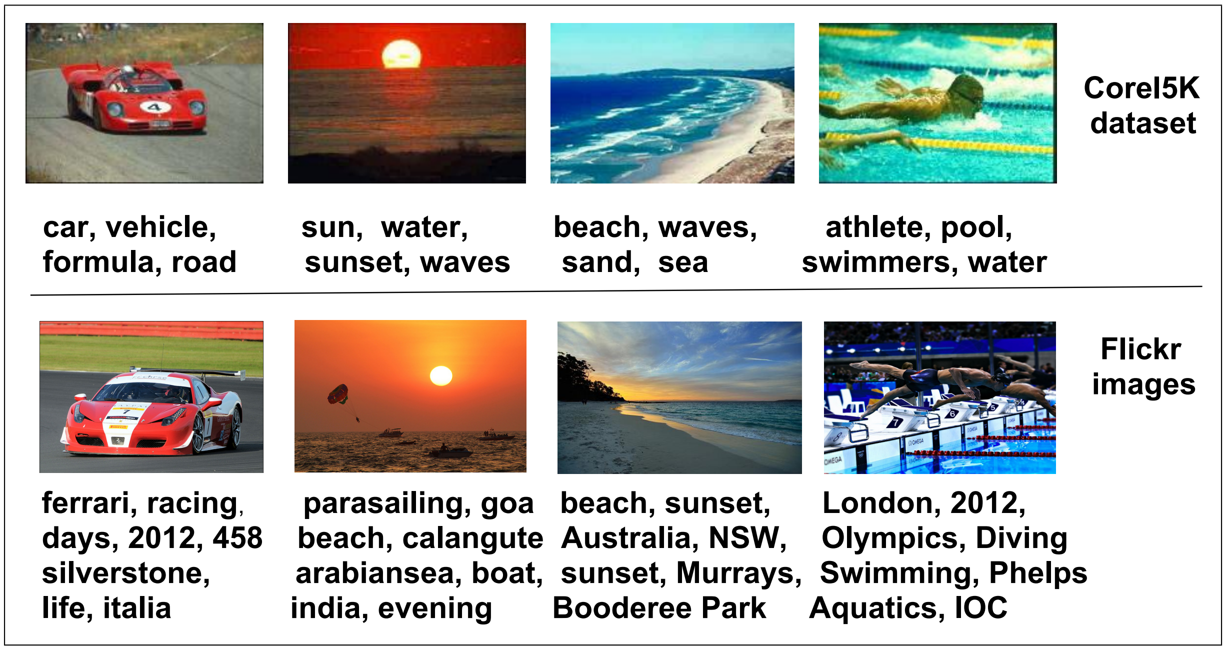

Corel5K dataset
car, vehicle, formula, road
sun, water, sunset, waves
beach, waves, sand, sea
athlete, pool, swimmers, water
Flickr images
ferrari, racing, days, 2012, 458 silverstone, life, italia
parasailing, goa beach, calangute arabiansea, boat, india, evening
beach, sunset, Australia, NSW, sunset, Murrays, Booderee Park
London, 2012, Olympics, Diving Swimming, Phelps Aquatics, IOC

Figure 6.2 Figure shows example images and corresponding labels from both Corel5K datasets and Flickr images.

6.4 Noisy Labels

In photo sharing websites, such as Flickr and Picasa, the labels of the images are collectively annotated by large group of heterogeneous users. It is believed that most of the labels are correct, although there are many incorrect and redundant labels. Figure 6.2 shows examples of images from both expert-annotated Corel5K dataset and user tagged Flickr images. Labels of Flickr images like *love, life, emotions, excited etc.* are large in number and also irrelevant to the image content, whereas labels of Corel5K are often small in number and precisely describe the content of the image. It can be observed that around 40-50% of the labels of Flickr images are irrelevant and are out-of context of the concept which the image represents. In this chapter, we address this problem of image annotation in presence of noisy labels.

Previously, both generative models [25] and supervised classification methods [29, 15] have been applied to improve the performance of image annotation. Such methods rely heavily on the quality of the training set. In the web image annotation, one feasible way to obtain sufficient training data is to parse the web content automatically. However, due to the well-known problem of complexity and variety of the web pages, it is difficult to keep the quality of the training data high. Although some efforts have been made to parse the web content intelligently, such as [14, 13]. However, the light weight methods, such as DOM, SAX are more widely used in real applications. Due to their unsupervised acquiring processes, the training set obtained automatically is usually impurer than what is required for traditional annotation problems

Existing annotation methods [61, 97, 101, 36] consider the labels associated with the images to be devoid of errors and belonging to a small fixed vocabulary, and hence, can be directly used for designing annotation schemes. They make an inherent assumption that labels present in the training set are reliable and correct, and hence can be directly used for training. In contrast, the labels collected by collaborative tagging websites are noisy i.e misspelled, redundant, irrelevant to content, and unlimited in numbers. These methods do not have an implicit mechanism of handling noisy labels and would not be suitable for annotation task in collaborative systems. Thus, an interesting problem to address is, on how to use the noisy information available for annotating unlabelled images reliably.

6.5 Image Annotation in Presence of Noisy Labels

We address the task of image annotation on noisy data using concept-modelling, a very popular notion in Information Retrieval community. The intuition is that, a specific meaning or aspect of an image can be well described by a group of highly related labels, referred to as label concept. Accordingly, each image can be organized into groups, each of which matches one label concept. This type of image organization not only removes noisy labels associated with an image, but also predicts additional labels that are actually depicted in the image but missing in the ground-truth annotations.

Methods like WSABIE [101], which learn a low dimension embedding space for images and annotations, address this issue in an indirect way. Even Wordnet-based approach [42] has been used to remove irrelevant labels. In MLFDA [99], image annotation is posed as a multi-modal multi-class classification problem, where the noisy data is treated as a special kind of modal of the class and separating hyperplanes between classes are learned by kernel-based LFDA.

In this chapter, we first present a graph-based approach for exemplifying the relationships between labels along with a noise removal algorithm to remove most of the semantically-unrelated links among the labels. We then make use of this label network to infer the semantic concepts associated with images. Finally we illustrate how these concepts could be used for image annotation in a KNN-based setting. Figure 6.3 summarizes our approach.

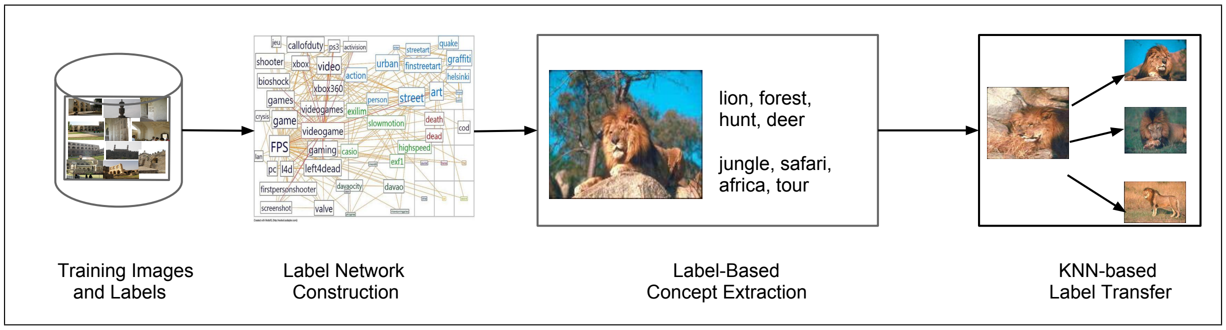

lion, forest, hunt, deer
jungle, safari, africa, tour
Training Images and Labels
Label Network Construction
Label-Based Concept Extraction
KNN-based Label Transfer

Figure 6.3 An overview of our approach, which includes label network construction based on their co-occurrence, semantic concept identification using image labels and a KNN-based approach for transferring labels of concepts to unannotated image.

To show the utility of *concepts* over noisy systems, we compare its annotation performance with a baseline annotation method JEC [61], with noisy labels on Corel5K and ESP datasets. Our experimental results suggest that the proposed concept-based method leads to superior image annotation performance compared to JEC in presence of noisy labels.

6.5.1 Label Network Construction and Noise Removal

The label network generation process along with noise removal step is very similar to the entity-network generation procedure described in section 3.1. Here, we briefly describe the process in the context of image labels and semantic concepts associated with them.

For label network creation, first the label co-occurrence counts, $\psi(\alpha, \beta)$ $\alpha, \beta \in V$, are calculated. But, this is not the best measure to quantify label co-occurrence strength as it may happen that two very frequent but uncorrelated tags might co-occur a lot compared to two relatively rare but correlated tags. Hence, we use *consistency*, $\phi(\alpha, \beta)$, to quantify associativity between labels, which is loosely defined as how much more likely is it to see the two labels together than random chance. We start by computing three types of raw statistics from the labels of training images : (i) Co-occurrence Counts $\psi(\alpha, \beta)$, (ii) Marginal Counts $\psi(\alpha)$, and (iii) Total Counts ψ_0 (defined in Step 3 of Algorithm 1). Joint probabilities $P(\alpha, \beta) = \frac{\psi(\alpha,\beta)}{\psi_0}$ and marginal probabilities $P(\alpha) = \frac{\psi(\alpha)}{\psi_0}$ are computed from these counts. These statistics are used for computing pair-wise consistencies between labels. We use *Normalized Point-Wise Mutual Information*[1] [9], defined as

$$\phi(\alpha, \beta) = \frac{\log\left(\frac{P(\alpha,\beta)}{P(\alpha)P(\beta)}\right)}{-\log P(\alpha, \beta)} \forall \psi(\alpha, \beta) > 0 \tag{6.2}$$

to exemplify this consistency between labels, as it is a well-bounded quantity and suitably satisfies the definition of consistency.

Iterative Noise Removal : Initially, there is insufficient knowledge to identify which label-pairs are noise. After computing consistencies, label pairs with consistencies lower than a threshold θ_{consy} can be declared noise and are removed from the network. The marginal and total counts are then updated and consistencies are recomputed in the next iteration. The label network generation algorithm along with the iterative noise removal method is described in Algorithm 1.

Table 6.1 shows effect of noise removal on the pair-wise consistencies of labels associated with label *water* in Corel5K data. It can be seen that consistencies of label *water* with correlated labels like *sea, ocean, beach, lake* increases significantly, whereas with irrelevant labels like *hills, grass* decreases to zero. By the end of this phase, we obtain a clean **noise-free** label network with pair-wise consistencies between labels as the edge weights, which we will call the *label consistency network*.

[1]http://en.wikipedia.org/wiki/Pointwise_mutual_information

Label	**sea**	**ocean**	**beach**	**lake**	**pool**	*hills*	*grass*
Before Denoising	0.127	0.172	0.219	0.1099	0.017	0.118	0.126
After Denoising	0.275	0.378	0.468	0.263	0.259	0	0

Table 6.1 Effect of noise removal on the consistencies of labels associated with label **water** in noisy Corel5K dataset, with θ_{consy} = 0.01.

6.5.2 Label-based Concept Extraction

Here, we use the label consistency network for identifying semantic concepts associated with training images, using the image labels as seed. We define concepts as local maximal subgraphs in the label consistency network, based on a novel measure of *concept strength*, which is in-turn defined in terms of *label strength*.

In a systematic way, we first define *label strength* of a node (label) in a subgraph as a measure that captures the connectivity of the node with rest of the nodes in the subgraph. This essentially is the eigenvector centrality [75] of the subgraph. If **x** = $\{x_1, x_2, ..., x_m\}$ be a set of m nodes in a subgraph and **W**(**x**) = $[\phi(x_i, x_j)]$ be the label consistency submatrix associated with this subgraph, then by eigenvector centrality, the *label strengths* converge to the first *unnormalized* eigenvector of **W**(**x**).

If $\lambda_1(\mathbf{W}(\mathbf{x}))$ is the first eigenvalue and $\mathbf{v}_1(\mathbf{W}(\mathbf{x}))$ is the first (normalized) eigenvector of this matrix, then *label strengths*, $\rho(\mathbf{x}|\mathbf{W}(\mathbf{x}))$, are defined by :

$$\rho(\mathbf{x}|\mathbf{W}(\mathbf{x})) = \lambda_1(\mathbf{W}(\mathbf{x})) \times \mathbf{v}_1(\mathbf{W}(\mathbf{x})) \tag{6.3}$$

$$\pi(\mathbf{x}|\Phi) = \min_{i=1...m} \{\rho_i\} \tag{6.4}$$

To capture the *tightness* of an arbitrary subgraph, we define *concept strength*, $\pi(\mathbf{x}|\Phi)$, to be **minimum** of the *label strengths* of all nodes (labels) of the subgraph (Equation 6.4). We now define *concepts*, as all those subgraphs in the label network, whose *concept strength* is higher than all its “neighbours”. Neighbours of a subgraph, **x**, is defined as all the subgraphs which can be obtained either by adding a single node ($\mathcal{N}_+(\mathbf{x})$) or removing a single node ($\mathcal{N}_-(\mathbf{x})$) from the given subgraph.

$$\mathcal{N}_+(\mathbf{x}) = \{\mathbf{y} = v \oplus \mathbf{x} | \forall v \in \mathbf{V} \backslash \mathbf{x}\} \qquad \mathcal{N}_-(\mathbf{x}) = \{\mathbf{y} = \mathbf{x} \backslash v | \forall v \in \mathbf{x}\} \tag{6.5}$$

We propose a greedy label-based approach to find such concepts, using the two atomic operations of **grow** and **shrink**. The **grow** operation tries to exhaustively find the best subgraph in $\mathcal{N}_+(\mathbf{x})$, which will have maximum *concept strength*, whereas the **shrink** operation finds the best subgraph in $\mathcal{N}_-(\mathbf{x})$.

Algorithm 4 explains how we extract multiple concepts associated with an image, using their labels as seed. The algorithm iterates over two phases : (i) **Shrink Phase**, which reduces labelset into a candidate subset of highly correlated labels, and (ii) **Grow Phase**, which adds more correlated labels to the candidate set making it a complete concept. The residue of the shrink phase is then again used as input over the next iteration to identify more concepts. Over this recursive process, multiple concepts associated with an image are identified.

Algorithm 4 Label Based Concept Extraction($\mathbf{x}_0$, Φ)

```
1: $\mathbf{x} \leftarrow \mathbf{x}_0$
2: $\mathbf{x}_{conc}$ = [ ]
3: while $\mathbf{x}$ do
4:     [$\mathbf{x}_{cand}, \mathbf{x}_{rem}$] = ShrinkPhase($\mathbf{x}|\Phi$)
5:     $\mathbf{x}_{conc}$ = [ $\mathbf{x}_{conc}$ GrowPhase($\mathbf{x}_{cand}|\Phi$) ] {Concepts extracted are concatenated}
6:     $\mathbf{x} \leftarrow \mathbf{x}_{rem}$
7: end while
8: A. ShrinkPhase($\mathbf{x}|\Phi$)
9: loop
10:     $[\mathbf{x}^-, \mathbf{x}_{rem}] \leftarrow Shrink(\mathbf{x}|\Phi)$ {Best possible down-neighbor.}
11:     if $\pi(\mathbf{x}^-) > \pi(\mathbf{x})$ then
12:         $\mathbf{x} \leftarrow \mathbf{x}^-$ {Not reached local maxima yet.}
13:     else
14:         return [$\mathbf{x}$, $\mathbf{x}_{rem}$] {Reached local maxima.}
15:     end if
16: end loop
17: B. GrowPhase($\mathbf{x}|\Phi$)
18: $\mathbf{x}^+ \leftarrow Grow(\mathbf{x}|\Phi)$ {Best possible up-neighbor.}
19: while $\pi(\mathbf{x}^+) > \pi(\mathbf{x})$ do
20:     $\mathbf{x} \leftarrow \mathbf{x}^+$
21:     $\mathbf{x}^+ \leftarrow Grow(\mathbf{x}|\Phi)$
22: end while
23: return $\mathbf{x}^+$
```

6.5.3 Label Transfer for Annotation

Now, we illustrate how *concepts* can be used for the task of image annotation algorithm in an NN-setting. As a pre-processing step, we first use the training image labels to create a label consistency network and concepts associated with individual training images are extracted.

Given a test image J, we first find the top K-visually similar training images using features and distance metrics discussed below:

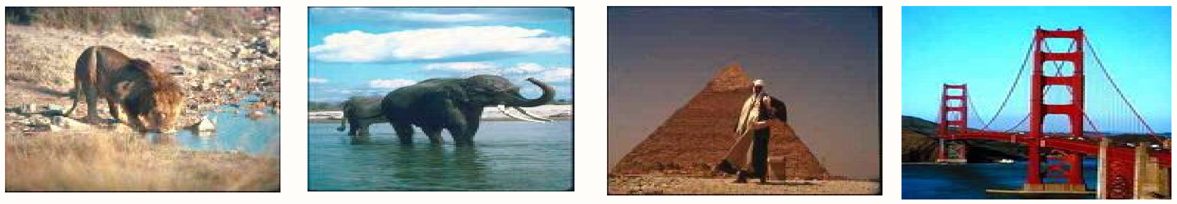
Features and Distances: We extract different types of features commonly used for image search and categorisation. We use two types of global image descriptors: Gist features [73], and color histograms with 16 bins in each color channel for RGB, LAB, HSV representations. Local features include SIFT as well as a robust hue descriptor [100], both extracted densely on a multiscale grid or for Harris-Laplacian interest points. Each local feature descriptor is quantized using k-means on samples from the training set. Images are then represented as a bag-of-words histogram. All descriptors but Gist are L_1-normalised and also computed in a spatial arrangement [51]. We compute the histograms over three horizontal regions of the image, and concatenate them to form a new global descriptor, albeit one that encodes some information of the spatial layout of the image. To limit color histogram sizes, here, we reduced the quantization to 12 bins in each channel. Note that this spatial binning differs from segmented image regions, as used in some previous work. This results in 15 distinct descriptors, namely one Gist descriptor, 6 color histograms and 8 bag-of-features (2 features types x 2 descriptors x 2 layouts). To compute the distances from the descriptors we follow previous work and use L_2 as the base metric for Gist, L_1 for global color histograms, and χ^2 for the others.

Combining Distances: Distance from different descriptors are combined using Joint Equal Contribution (JEC) framework, as proposed in [61]. Since, in JEC, each feature contributes equally towards the image distance, we first need to find the appropriate scaling terms for each feature. These scaling terms can be determined easily if the features are normalized in some way (e.g., features that have unit norm), but in practice this is not always the case. We can obtain estimates of the scaling terms by examining the lower and upper bounds on the feature distances computed on some training set. We scale the distances for each feature such that they are bounded by 0 and 1. If we denote the scaled distance as $\widetilde{d}^k_{(i,j)}$, we can define the comprehensive image distance between images I_i and I_j as $\sum\limits_{k=1}^{N} \frac{\widetilde{d}^k_{(i,j)}}{N}$. We refer to this distance as Joint Equal Contribution (JEC).

Then, the labels associated with the concepts of the nearest training images are ranked based on a label scoring function, κ_{J,v_i}, defined as :

$$\kappa_{J,v_i} = \gamma_{J,K} \cdot \pi(\mathbf{x}|\Phi) \cdot \rho_{v_i}(\mathbf{x}) \tag{6.6}$$

This score is computed as product of visual similarity of the training image to the test image, concept strength of the concept associated with the training image and the label strength of the label within the concept. The individual components of the scoring function are first normalized before computing the scores. The labels are ranked based on this score and top L unique labels are assigned to the test image.



lion, water, grass, forest	elephant, river, trunk	desert, pyramid, sun, sand,	bay, water, bridge
lion, grass, forest, birds, hills	elephant, river, water, sky, clouds	pyramid, sun, desert. beach, sunrise	water, ocean, sea bridge, clouds
lion, water, grass, forest, river	elephant, clouds, river, sky, trunk	pyramid, desert, sand, sun, dunes	sea, water, bridge, bay, hills

Figure 6.4 Annotation of test images from noisy Corel5K dataset. The second, third and fourth rows show the ground truth labels, the labels predicted by JEC and the labels predicted by our method respectively. The labels in red are those, though depicted in the corresponding images, are missing in the ground-truth annotations and are predicted by our method.

Please note same label could have multiple scores, due to presence of same label in multiple concepts or images.

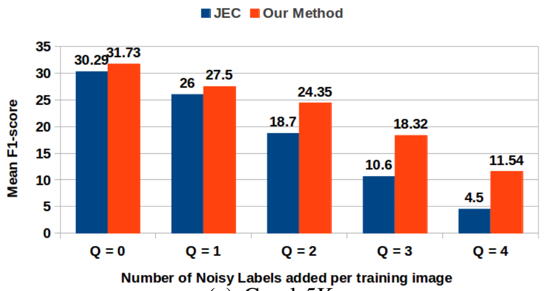
6.6 Experiments

We present both qualitative and quantitative results, showing comparisons of our method with a very popular baseline method JEC [61], on benchmark annotation datasets : Corel5K and ESP [61].

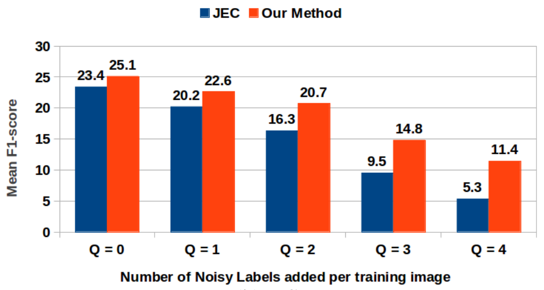
- Corel5K [22] has become a de-facto evaluation benchmark in the image annotation community. It contains 5,000 images collected from the larger Corel CD set, split into 4,500 training and 500 test examples. Each image is annotated with an average of 3.5 keywords, and the dictionary contains 260 words that appear in both the train and the test set.

- ESP Game [2] consists of a set of 21,844 images collected in the ESP collaborative image labeling task. In ESP game [108], two players assign labels to the same image without communicating. Only common labels are accepted. As an image is shown to more teams, a list of taboo words is accumulated, increasing the difficulty for future players and resulting in a challenging dataset for annotation. The set we obtained [3] contains a wide variety of images annotated by 269 keywords, and is split into 19,659 train and 2,185 test images. Each image is associated with up to 15 keywords, and on average 4.6 keywords.

[2]http://www.espgame.org
[3]http://hunch.net/ ji/



(a) Corel-5K



(b) ESP

Figure 6.5 Comparison of annotation performance of our method and JEC[61] on noisy Corel5K and ESP datasets. Q denotes the number of noisy labels added per training image.

As the motivation of our work is to show the effectiveness of our method on data with noisy labels, we create a parameter modulated noisy dataset by adding noisy labels to the training images of Corel5K and ESP. The noisy labels are taken from a vocabulary which has no overlap with the ground-truth vocabularies. We perform modulated experiments by regulating the degree of noise added to training data, using a parameter Q, which denotes the number of noisy labels added per training image. It is ensured that there is no overlap between the candidate noisy tags and dictionary of the dataset. Annotation models are created using both, our method and JEC [61]. Evaluations are done using popular metric of mean F1-score over all the labels in the original vocabulary of the dataset. F1-score is defined as the harmonic mean of the mean precision and recall rates obtained by different models. Precision and recall are defined in the standard way: the annotation precision for a keyword is defined as the number of images assigned the keyword correctly divided by the total number of images predicted to have the keyword. The annotation recall is defined as the number of images assigned the keyword correctly, divided by the number of images assigned the keyword in the ground-truth annotation. Similar to other approaches, we assign top 5 keywords to each image using label transfer. The F1-scores reported by our method correspond to label networks with threshold $\theta_{consy} = 0.01$, which was experimentally observed to be giving best results.

Figure 6.4 shows some qualitative results obtained using our method and JEC on noisy Corel5K data. The second row shows the ground-truth annotations, while the third and forth rows show the labels predicted by JEC and our method respectively. It can be seen that some labels predicted by JEC are irrelevant and, also some ground truth annotations are missing in the predictions, whereas our method predicts all ground-truth annotations along with labels, which are depicted in the image but missing in the ground-truth annotation.

To analyze our method's performance quantitatively, we compute the F1-score of each label in the ground-truth. The mean F1 scores using our method as well as those obtained by JEC [61] are reported in Figure 6.5. In both Corel5K and ESP datasets, as noise increases, F1-score of both the methods decrease. It can also be seen that as the degree of noise (Q) increases, the relative performance of our method, in comparison to JEC, increases. In case of Corel5K, when only one noisy label is added per training image (Q = 1), there is about 6% improvement in F1-score. As we increase Q to 4, there is around 150% increase in the F1-score, which is a very significant improvement. This shows the effectiveness of using *concepts* in the task of image annotation, especially when the amount of noise is too high.

Experimentally we found that as θ_{consy} increases, the F1 scores also increase upto to a saturation point, and then start decreasing. This happens because once θ_{consy} reaches its saturation value, even relevant label-pairs in the network are considered as noise and discarded in the noise removal step. Our method takes more processing time compared to JEC, which typically has negligible training time.

6.7 Summary

In this chapter, we propose a novel knowledge-based approach for image annotation that exploits the semantic label concepts, derived based on the collective knowledge embedded in label co-occurrence based consistency network. We first define the notion of *concept strength* of a subgraph as a measure which captures the connectivity of nodes within the subgraph, by introducing a novel notion of *concept strength* of a subgraph, as a indirect function of the edge weights of the subgraph. *Local Concepts* are defined as those subgraphs whose *concept strength* is higher than all its *neighbors*. Our proposed semantic concept based framework is generic in nature. In particular, the noise removal stage during the label network construction and the strict notion of defining concepts allowed us to perform well in noisy datasets. An important future work to pursue is to build a *hierarchy of Concepts* and utilize them to learn useful insights for the tasks of image annotation and retrieval.

Chapter 7

Conclusions

In this thesis, we presented CoocMiner, an end-to-end framework for mining socila media data. It generates highly noise-robust weighted entity co-occurrence network addressing the fundamental mixture of concept properties in entity-set data and mines “tight” overlapping communities from this weighted network using novel notions of coherence as community strength and soft maximal cliques as communities. CoocMiner has very few highly interpretable parameters, it can scale to very large networks due to the inherent parallelism in the way seed based community detection methods work and the nature of the greedy algorithm it uses. The joint effect of the aggressive noise removal and use of weights in defining and detecting communities leads to the discovery of a large number of small but very meaningful communities compared to a small number of large and noisy communities obtained by traditional methods that become even worse as we apply them to more noisy datasets (e.g. Flickr, Wikipedia). Moreover, CoocMiner significantly outperforms other community detection methods and LDA in an application-oriented task of tag recommendation.

We also present an algorithm for identifying large and loose communities discovered by any community detection method and partition them into compact and meaningful communities, if needed. We use coherence for defining compactness of community and propose iterative method for partitioning loose communities.

We also proposed a novel knowledge-based approach for image annotation that exploits the semantic label concepts, which are derived based on the collective knowledge embedded in label co-occurrence based consistency network.

In the future CoocMiner can also be extended to discovering a hierarchy of communities by annotating the entity-set data at each level with the communities found and then applying the same algorithm to find co-occurrences among communities at the next level. The other extension of CoocMiner is for weighted tagsets such as Bag-of-words in text data.

Related Publications

1. **Logical Itemset Mining**
   Shailesh Kumar, Chandrashekar V & C V Jawahar.
   Workshop Proceedings of International Conference on Data Mining, Brussels, Belgium, December 2012.

2. **Compacting Large and Loose Communities**
   Chandrashekar V, Shailesh Kumar & C V Jawahar.
   Asian Conference on Pattern Recognition, Okhinawa, Japan, November 2013.

3. **Image Annotation in Presence of Noisy Labels**
   Chandrashekar V, Shailesh Kumar & C V Jawahar.
   International Conference on Pattern Recognition and Machine Intelligence, Kolkata, India, December 2013.

4. **Mining Logical Concepts in Entity Co-occurrence Graphs using Coherent Community Detection** (Under Review)
   Shailesh Kumar, Chandrashekar V & C V Jawahar.
   IEEE Transactions on Knowledge and Data Engineering, August 2013.

Bibliography

[1] E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne. Finding high-quality content in social media. In *WSDM*, pages 183–194, 2008.

[2] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *SIGMOD*, pages 207–216, 1993.

[3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules in large databases. In *VLDB*, pages 487–499, 1994.

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

[5] R. A. Baeza-Yates. Graphs from Search Engine Queries. In *SOFSEM 2007: Theory and Practice of Computer Science*, volume 4362/2007, pages 1–8, 2007.

[6] G. Begelman. Automated tag clustering: Improving search and exploration in the tag space. In *In Proc. of the Collaborative Web Tagging Workshop at WWW06*, 2006.

[7] D. M. Blei and M. I. Jordan. Modeling annotated data. In *SIGIR*, page 127134, 2003.

[8] C. Borgelt. Recursion pruning for the apriori algorithm. In *FIMI*, 2004.

[9] G. Bouma. Normalized (pointwise) mutual information in collocation extraction. *Biennial GSCL*, 2009.

[10] R. L. Breiger. The duality of persons and groups. *Social forces*, 53(2):181–190, 1974.

[11] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[12] C. Bron and J. Kerbosch. Algorithm 457: finding all cliques of an undirected graph. *Commun. ACM*, 16(9):575–577, 1973.

[13] D. Cai, X. He, Z. Li, W.-Y. Ma, and J.-R. Wen. Hierarchical clustering of www image search results using visual, textual and link information. In *ACM Multimedia*, pages 952–959, 2004.

[14] D. Cai, S. Yu, J. rong Wen, W. ying Ma, D. Cai, S. Yu, J. rong Wen, and W. ying Ma. Vips: a vision-based page segmentation algorithm, 2003.

[15] G. Carneiro. Formulating semantic image annotation as a supervised learning problem. In *CVPR*, 2005.

[16] C. Cattuto, D. Benz, A. Hotho, and G. Stumme. Semantic grounding of tag relatedness in social bookmarking systems. In *ISWC*, pages 615–631, 2008.

[17] J. Chen, O. R. Zaïane, and R. Goebel. A visual data mining approach to find overlapping communities in networks. In *ASONAM*, pages 338–343, 2009.

[18] A. Clauset. Finding local community structure in networks. *Physical Review*, E(72), 2005.

[19] A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Physical Review*, E(70), 2004.

[20] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, New York, NY, USA, 1991.

[21] J. Duch and A. Arenas. Community detection in complex networks using extremal optimization. *Physical Review*, E(72), 2005.

[22] P. Duygulu, K. Barnard, J. F. G. d. Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *ECCV*, pages 97–112, 2002.

[23] I. J. Farkas, D. Abel, G. Palla, and T. Vicsek. Weighted network modules. *New Journal of Physics*, 186(9), 2007.

[24] S. L. Feng, R. Manmatha, and V. Lavrenko. Multiple bernoulli relevance models for image and video annotation. In *CVPR*, pages 1002–1009, 2004.

[25] S. L. Feng, R. Manmatha, and V. Lavrenko. Multiple bernoulli relevance models for image and video annotation. In *CVPR*, pages 1002–1009, 2004.

[26] G. W. Flake, S. Lawrence, and C. L. Giles. Efficient identification of web communities. *SIGKDD*, pages pp 150–160, 2000.

[27] S. Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75174, 2010.

[28] S. Fortunato and M. Barthélemy. Resolution limit in community detection. *PNAS*, 104(1):36–41, 2007.

[29] Y. Gao, J. Fan, X. Xue, and R. Jain. Automatic image annotation by incorporating feature hierarchy and boosting to scale up svm classifiers. In *ACM Multimedia*, pages 901–910, 2006.

[30] J. Gemmell, A. Shepitsen, B. Mobasher, and R. Burke. Personalizing navigation in folksonomies using hierarchical tag clustering. *Proceedings of DaWaK*, pages 196–205, 2008.

[31] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys*, 38(3), 2006.

[32] E. Giannakidou, V. A. Koutsonikola, A. Vakali, and Y. Kompatsiaris. Co-clustering tags and social data sources. *Proceedings of WAIM*, page 317324, 2008.

[33] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *PNAS*, 99(12):7821–7826, 2002.

[34] S. Gregory. Local betweenness for finding communities in networks. Technical report, University of Bristol, 2008.

[35] S. Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12, 2010.

[36] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. Tagprop: Discriminative metric learning in nearest neighbor models for image auto-annotation. *ICCV*, pages 309–316, 2009.

[37] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *SIGMOD Conference*, pages 1–12, 2000.

[38] J. Hare, P. Lewis, P. Enser, J. Christine, and O. B. Mind the gap: Another look at the problem of the semantic gap in image retrieval. In *Proceedings of Multimedia Content Analysis, Management and Retrieval*, 2006.

[39] Y. M. Hironobu, H. Takahashi, and R. Oka. Image-to-word transformation based on dividing and vector quantizing images with words. In *in Boltzmann machines, Neural Networks*, pages 405–409, 1999.

[40] C.-J. Hsieh and I. S. Dhillon. Fast coordinate descent methods with variable selection for non-negative matrix factorization. In *KDD*, pages 1064–1072, 2011.

[41] J. Jeon, V. Lavrenko, and R. Manmatha. Automatic image annotation and retrieval using cross-media relevance models. In *SIGIR*, pages 119–126, 2003.

[42] Y. Jin, L. Khan, L. Wang, and M. Awad. Image annotations by combining multiple evidence and wordnet. *ACM Multimedia*, 2005.

[43] A. M. Kaplan and M. Haenlein. Users of the world, unite! The challenges and opportunities of Social Media. *Business Horizons*, 53(1):59–68, 2010.

[44] G. Karypis and V. Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20(1):359–392, 1998.

[45] J. H. Kietzmann, K. Hermkens, I. P. McCarthy, and B. S. Silvestre. Social media? get serious! understanding the functional building blocks of social media. *Business Horizons*, 54(3):”241 – 251, 2011.

[46] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. In *Computer Networks*, pages 1481–1493, 1999.

[47] A. Lancichinetti, S. Fortunato, and J. Kertsz. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11, 2009.

[48] A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato. Finding statistically significant communities in networks. *PLoS ONE*, 6(5), 2011.

[49] S. Lattanzi and D. Sivakumar. Affiliation networks. In *STOC*, pages 427–434, 2009.

[50] V. Lavrenko, R. Manmatha, and J. Jeon. A model for learning the semantics of pictures. In *NIPS*, 2003.

[51] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, pages 2169–2178, 2006.

[52] C. Lee, F. Reid, A. McDaid, and N. Hurley. Detecting highly overlapping community structure by greedy clique expansion. *Workshop - ACM KDD-SNA*, 2010.

[53] D. D. Lee and H. S. Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401:788–791, 1999.

[54] X. Li, G. M. S. Cees, and M. Worring. Learning social tag relevance by neighbor voting. *IEEE Transactions on Multimedia*, 11(7):1310–1322, 2009.

[55] X. Li, C. G. M. Snoek, and M. Worring. Learning social tag relevance by neighbor voting. *IEEE Transactions on Multimedia*, 11(7):1310–1322, 2009.

[56] C.-J. Lin. Projected gradient methods for nonnegative matrix factorization. *Neural Computation*, 19(10):2756–2779, 2007.

[57] Y.-R. Lin, J. Sun, P. Castro, R. B. Konuru, H. Sundaram, and A. Kelliher. Metafac: community discovery via relational hypergraph factorization. In *KDD*, pages 527–536, 2009.

[58] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[59] C. Lucchese, S. Orlando, and R. Perego. kdci: on using direct count up to the third iteration. In *FIMI*, 2004.

[60] F. Luo, J. Z. Wang, and E. Promislow. Exploring local community structures in large networks. *Web Intelligence*, pages 233–239, 2006.

[61] A. Makadia, V. Pavlovic, and S. Kumar. Baselines for image annotation. *IJCV*, 90(1):88–105, 2010.

[62] C. P. Massen and J. P. K. Doye. Identifying communities within energy landscapes. *Physical Review*, E(71), 2005.

[63] D. Metzler and R. Manmatha. An inference network approach to image retrieval. In *CIVR*, pages 42–50, 2004.

[64] P. Mika. Ontologies are us: A unified model of social networks and semantics. In *International Semantic Web Conference*, pages 522–536, 2005.

[65] F. Monay and D. Gatica-Perez. On image auto-annotation with latent space models. In *ACM Multimedia*, pages 275–278, 2003.

[66] M. E. Newman. Modularity and community structure in networks. *PNAS*, 103(23):8577–8582, 2006.

[67] M. E. J. Newman. Analysis of weighted networks. *Physical Review*, E(70), 2004.

[68] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Physical Review*, E(69), 2004.

[69] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review*, E(74), 2006.

[70] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review*, E(69), 2004.

[71] V. Nicosia, G. Mangioni, V. Carchiolo, and M. Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Stat. Mech.*, 2009.

[72] Nielsen. State of the media: The social media report, 2012.

[73] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.

[74] J. P. Onnela, J. Saramki, J. Kertsz, and K. Kaski. Intensity and coherence of motifs in weighted complex networks. *Physical Review E*, 71, 2005.

[75] T. Opsahl, F. Agneessens, and J. Skvoretz. Node centrality in weighted networks: Generalizing degree and shortest paths. *Social Networks*, 32(3):245–251, 2010.

[76] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

[77] S. Papadopoulos, Y. Kompatsiaris, and A. Vakali. A graph-based clustering scheme for identifying related tags in folksonomies. *Proceedings of DaWaK*, pages 65–76, 2010.

[78] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, and P. Spyridonos. Community detection in social media. *Data Mining and Knowledge Discovery*, 24(3):515–554, 2012.

[79] X.-H. Phan and C.-T. Nguyen. Gibbslda++: A c/c++ implementation of latent dirichlet allocation (lda), 2007.

[80] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Dening and identifying communities in networks. *PNAS*, 101(9), 2004.

[81] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3), 2007.

[82] S. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[83] R. Schifanella, A. Barrat, C. Cattuto, B. Markines, and F. Menczer. Folks in folksonomies:social link prediction from shared metadata. *Proceedings of WSDM*, page 271280, 2010.

[84] C. Schmitz, A. Hotho, R. Jschke, and G. Stumme. Mining association rules in folksonomies. In *Data Science and Classification: Proc. of the 10th IFCS Conf.*, pages 261–270, 2006.

[85] J. Scripps, P.-N. Tan, and A.-H. Esfahanian. Node roles and community structure in networks. In *WebKDD/SNA-KDD*, pages 26–35, 2007.

[86] H. Shen, X. Cheng, K. Cai, and M.-B. Hu. Detect overlapping and hierarchical community structure in networks. *CoRR*, abs/0810.3093, 2008.

[87] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE PAMI*, 22(8):888905, 2000.

[88] Z. Shi, H. Rui, and A. B. Whinston. Content sharing in a social broadcasting environment: Evidence from twitter, 2013.

[89] E. Simpson. Clustering tags in enterprise and web folksonomies. *Technical Report HPL*, 2008.

[90] L. Specia and E. Motta. Integrating folksonomies with the semantic web. *Proceedings of the 4th European Conference on the Semantic Web : Research and Applications*, pages 624–639, 2007.

[91] L. Szathmary, A. Napoli, and P. Valtchev. Towards rare itemset mining. In *ICTAI (1)*, pages 305–312, 2007.

[92] L. Szathmary, P. Valtchev, and A. Napoli. Finding minimal rare itemsets and rare association rules. In *KSEM*, pages 16–27, 2010.

[93] P. N. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. *Proceedings of SIGKDD*, pages 32–41, 2002.

[94] Q. Tang, B. Gu, and A. Whinston. Content contribution for revenue sharing and reputation in social media: A dynamic structural model. *J. Manage. Inf. Syst.*, 29(2):41–76, 2012.

[95] D. Tsatsou, P. Symeon, K. Ioannis, and P. C. Davis. Distributed technologies for personalized advertisement delivery. In *Online Multimedia Advertising: Techniques and Technologies*, pages 233–261, 2011.

[96] T. Uno, M. Kiyomi, and H. Arimura. Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In *FIMI*, 2004.

[97] Y. Verma and C. V. Jawahar. Image annotation using metric learning in semantic neighbourhoods. *ECCV*, pages 836–849, 2012.

[98] Q. Wan and A. An. Efficient mining of indirect associations using hi-mine. In *Canadian Conference on AI*, pages 206–221, 2003.

[99] M. Wang, X. Zhou, and H. Xu. Web image annotation based on automatically obtained noisy training set. *APWeb*, pages 637–648, 2008.

[100] J. V. D. Weijer and C. Schmid. Coloring local feature extraction. In *ECCV*, 2006.

[101] J. Weston, S. Bengio, and N. Usunier. Wsabie : Scaling up to large vocabulary image annotation. *IJCAI*, 2011.

[102] J. Xie, S. Kelley, and B. K. Szymanski. Overlapping community detection in networks: the state of the art and comparative study. *ACM Computing Surveys*, 45(4), 2013.

[103] J. Xie, B. K. Szymanski, and X. Liu. Slpa: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. *ICDM 2011 Workshop on DMCCI*, 2011.

[104] X. Xu, N. Y. Z. Feng, and T. A. J. Schweiger. Scan: A structural clustering algorithm for networks. *Proceedings of SIGKDD*, pages 824–833, 2007.

[105] J. Yang and J. Leskovec. Community-affiliation graph model for overlapping network community detection. In *ICDM*, pages 1170–1175, 2012.

[106] J. Yang and J. Leskovec. Defining and evaluating network communities based on ground-truth. In *ICDM*, pages 745–754, 2012.

[107] J. Yang and J. Leskovec. Overlapping community detection at scale: a nonnegative matrix factorization approach. In *WSDM*, pages 587–596, 2013.

[108] A. Yavlinsky, E. Schofield, and S. Rüger. Automated image annotation using global features and robust nonparametric density estimation. In *CIVR*, pages 507–517, 2005.

[109] C. M. Yeung, N. Gibbins, and N. Shadbolt. Contextualising tags in collaborative tagging systems. *Proceedings of ACM Conference on Hypertext and Hypermedia*, pages 251–260, 2009.

[110] M. J. Zaki. Scalable algorithms for association mining. *IEEE Trans. Knowl. Data Eng.*, 12(3):372–390, 2000.

[111] Q. Zhao, P. Mitra, and B. Chen. Temporal and information flow based event detection from social text streams. In *AAAI*, pages 1501–1506, 2007.