A Graph-Based Clustering Scheme for Identifying Related Tags in Folksonomies

Symeon Papadopoulos[1,2], Yiannis Kompatsiaris[1], and Athena Vakali[2]

[1] Informatics and Telematics Institute, CERTH
57001, Thessaloniki, Greece
{papadop,ikom}@iti.gr
[2] Department of Informatics, Aristotle University,
54124, Thessaloniki, Greece
avakali@csd.auth.gr

Abstract. The paper presents a novel scheme for graph-based clustering with the goal of identifying groups of related tags in folksonomies. The proposed scheme searches for core sets, i.e. groups of nodes that are densely connected to each other by efficiently exploring the two-dimensional core parameter space, and successively expands the identified cores by maximizing a local subgraph quality measure. We evaluate this scheme on three real-world tag networks by assessing the relatedness of same-cluster tags and by using tag clusters for tag recommendation. In addition, we compare our results to the ones derived from a baseline graph-based clustering method and from a popular modularity maximization clustering method.

Keywords: graph-based clustering, community detection, folksonomies, tag recommendation.

1 Introduction

Collaborative (or Social) Tagging is nowadays a common feature of content sharing web applications that enables users to: (a) upload new, or bookmark existing content and, (b) annotate it by means of free-text keywords (tags). Such applications, examples of which are delicious[1], flickr[2] and Bibsonomy[3], are commonly referred to as Social Tagging Systems (STS). Currently, STS attract huge amounts of traffic, which results in the emergence of massive grassroots content annotation and organization schemes, referred to as folksonomies [1,2]. Folksonomies comprise three types of entities, namely users, resources and tags, as well as the associations among them [3,4].

Folksonomies constitute a direct encoding of the views of a large number of users on how content items should be organized through a flexible annotation scheme (tagging). By analyzing the structure and content of folksonomies, one

[1] http://delicious.com/
[2] http://www.flickr.com/
[3] http://bibsonomy.org/

T.B. Pedersen, M.K. Mohania, and A M. Tjoa (Eds.): DaWaK 2010, LNCS 6263, pp. 65–76, 2010.

can expect to gain valuable insights into the topic and vocabulary structure of the system. To this end, *tag clustering* has lately attracted significant research interest due to its value in several Information Retrieval (IR) use case scenarios [5,6,7,8,9,10,11]. Tag clustering is commonly understood as a process that groups the tags of an STS in a way such that members of the same tag cluster are perceived by users as *related* to each other. Despite the subjective element in judging the degree of relatedness between tags, tag clusters are expected to correspond to meaningful topic areas, which can be useful in a series of tasks, such as information exploration and navigation [5,6], automatic content annotation [8], user profiling [9], content clustering [10,11] and tag recommendation [12,13].

To date, tag clustering has been dealt with either by conventional clustering algorithms, such as K-means [10] and Hierarchical Agglomerative Clustering [8,9], or, more recently, by use of *community detection* methods [5,6,7]. Conventional clustering schemes are frequently troubled by two shortcomings: (a) the need for providing the number of clusters as input to the algorithm, and (b) their computational complexity. Community detection methods address both of these needs, since they do not require the number of clusters (communities) to be known a priori and they are typically more efficient in terms of computations. However, *modularity maximization* methods [14], which constitute the bulk of community detection methods, are troubled by the so-called "super-community" problem, i.e. they produce few communities with very large sizes and numerous communities with small sizes. Having tag clusters of such highly skewed size distribution can be detrimental to the aforementioned IR tasks.

For that reason, we introduce in this paper a hybrid graph-based tag clustering scheme, referred to in short as HGC, which attempts to address the aforementioned constraints. HGC is based on the notion of (μ, ϵ)-cores [15], groups of nodes that have a large number of common neighbors to each other. HGC conducts an efficient search over the (μ, ϵ) parameter space and identifies the associated core sets. Subsequently, a core set expansion step is conducted based on a local modularity measure [16]. This expansion enables the resulting clusters to overlap with each other, which is particularly important for the problem of tag clustering, since tags are typically used in multiple contexts and senses.

The rest of the paper is structured as follows. Section 2 discusses existing work on the topic of tag clustering and its applications. Section 3 presents HGC, the proposed hybrid graph-based solution to the problem of tag clustering. HGC is evaluated and compared against existing clustering schemes in Section 4. The paper concludes in Section 5.

2 Related Work

The problem of tag clustering has recently attracted increasing research interest since it is a challenging task from a data mining perspective, but at the same time it also holds the potential for benefiting a variety of IR applications. For instance, tag clustering is considered important for eliciting a topic hierarchy for a tagging system and improving content retrieval and browsing [8]. Similar conclusions are reached by [5] who point that the use of raw tag information limits content exploration and discovery, thus creating the need for an additional level of organization through tag clustering. In [9], tag clusters are used as a nexus between users and their interests. Using tag clusters instead of plain tags for profiling user interests proved beneficial for personalized content ranking. An additional application of tag clustering is presented in [7]. There, the tag clusters were used as a means of identifying the different contexts of use for a given tag, i.e. for sense disambiguation. It was shown that using the tag clusters results in improved results compared to the use of external resources such as WordNet.

The methods used for performing the tag clustering largely fall under one of two approaches: (a) conventional clustering techniques, such as Hierarchical Agglomerative Clustering (HAC) [8,9] and (b) community detection methods [5,6,7]. HAC suffers from high complexity (quadratic to the number of tags to be clustered) and the need to set ad-hoc parameters (e.g. three parameters need to be set in the clustering scheme used in [9]). Community detection methods largely address the shortcomings of HAC since efficient implementations exist with a complexity of $O(Nlog(N))$ for finding the optimal grouping of N tags into communities. Furthermore, community detection methods rely on the measure of modularity [14] as a means to assess the quality of the derived cluster structure. Thus, modularity maximization methods do not require any user-defined parameters. However, a problem of modularity maximization methods pointed in [6] and confirmed by our experiments is their tendency to produce clusters with a highly skewed size distribution, which makes them unsuitable for the problem of tag clustering.

3 Description of HGC

The proposed scheme builds upon the notion of (μ, ϵ)-cores introduced in [15] and briefly described in subsection 3.1. The original algorithm, referred to as SCAN [15], suffers from two problems. First, it needs two parameters, namely μ and ϵ, to be provided as input. Second, it leaves a substantial number of nodes unassigned to clusters. As a result, its utility is limited in IR tasks such as tag recommendation. For that reason, our scheme conducts an efficient iterative search over the parameter space (μ, ϵ) in order to discover cores for multiple values of the parameters (subsection 3.2). Finally, the identified cores are expanded, as described in subsection 3.3, by maximizing a local measure of modularity [16] in order to increase the number of nodes that are assigned to communities and to enable overlap among communities.

3.1 Core Set Discovery

The definition of (μ, ϵ)-cores is based on the concepts of *structural similarity*, *ϵ-neighborhood* and *direct structure reachability*.

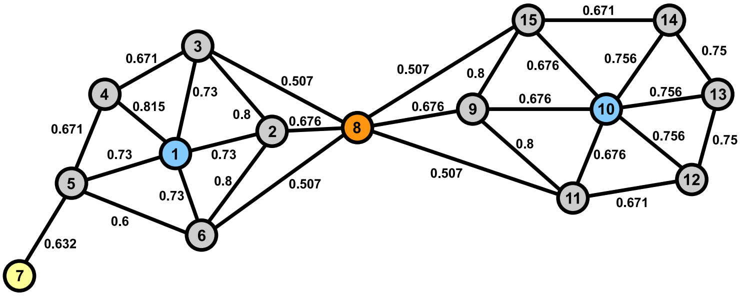

Fig. 1. Example of community structure in an artificial network. Nodes are labeled with successive numbers and edges are labeled with the structural similarity value between the nodes that they connect. Nodes 1 and 10 are (μ, ϵ)-cores with $\mu = 5$ and $\epsilon = 0.65$. Nodes 2-6 are structure reachable from node 1 and nodes 9, 11-15 are structure reachable from node 10. Thus, two community seed sets have been identified: the first consisting of nodes 1-6 and the second consisting of nodes 9-15.

Definition 1. *The* ***structural similarity*** *σ between two nodes v and w of a graph $G = \{V, E\}$ is defined as:*

$$\sigma(v, w) = \frac{|\Gamma(v) \cap \Gamma(w)|}{\sqrt{|\Gamma(v)| \cdot |\Gamma(w)|}} \tag{1}$$

where $\Gamma(v)$ is the structure *of node v: $\Gamma(v) = \{w \in V | (v, w) \in E\} \cup \{v\}$.*

Definition 2. *The* ***ϵ-neighborhood*** *of a node is the subset of its structure containing only the nodes that are at least ϵ-similar with the node; in math notation:*

$$N_\epsilon(v) = \{w \in \Gamma(v) | \sigma(v, w) \geq \epsilon\} \tag{2}$$

Definition 3. *A vertex v is called a* ***(μ, ϵ)-core*** *if its ϵ-neighborhood contains at least μ vertices: $CORE_{\mu,\epsilon}(v) \Leftrightarrow |N_\epsilon(v)| \geq \mu$.*

Definition 4. *A node is directly* ***structure reachable*** *from a (μ, ϵ)-core if it is at least ϵ-similar to it: $DirReach_{\mu,\epsilon}(v, w) \Leftrightarrow CORE_{\mu,\epsilon}(v) \wedge w \in N_\epsilon(v)$.*

Once the (μ, ϵ)-cores of a network have been identified, it is possible to start attaching adjacent nodes to them provided that they are reachable through a chain of nodes which are directly structure reachable from each other. We call the resulting set of nodes as a *community seed set*. The rest of the nodes are considered to be *hubs* or *outliers* depending on whether they are adjacent to more than one community core sets or not. An example of computing structural similarity values for the edges of a network and then identifying the underlying (μ, ϵ)-cores, hubs and outliers of the network is illustrated in Figure 1. This technique for collecting community seed sets is computationally efficient since its complexity is $O(\overline{k} \cdot n)$ for a network of n nodes and average degree $\overline{k}$. Computing the structural similarity values of the m network edges introduces an additional $O(\overline{k} \cdot m)$ complexity in the community detection.

3.2 Parameter Space Exploration

One issue that is not addressed in [15] pertains to the selection of parameters μ and ϵ. Setting a high value for ϵ (the maximum possible value for ϵ is 1.0) will render the core detection step very eclectic, i.e. few (μ, ϵ)-cores will be detected. Moreover, higher values for μ will also result in the detection of fewer cores (for instance, all nodes with degree lower than μ will be excluded from the core selection process). For that reason, we employ an iterative scheme, in which the community seed set selection operation is carried out multiple times with different values of μ and ϵ so that a meaningful subspace of these two parameters is thoroughly explored and the respective (μ, ϵ)-cores are detected.

The exploration of the (μ, ϵ) parameter space is carried out as depicted in Figure 2. We start by a very high value for both parameters. Since the maximum possible values for μ and ϵ are k_{max} (maximum degree on the graph) and 1.0 respectively, we start the parameter exploration by two values dependent on them (for instance, we could select $\mu_0 = 0.5 \cdot k_{max}$ and $\epsilon_0 = 0.9$; the results of the algorithm are not very sensitive to this choice). We identify the respective (μ, ϵ) cores and associated core sets and then relax the parameters in the following way. First, we reduce μ; if it falls below a certain threshold (e.g. $\mu_{min} = 4$), we then reduce ϵ by a small step (e.g. 0.05) and we reset $\mu = \mu_0$. When both μ and ϵ reach a small value ($\mu = \mu_{min}$ and $\epsilon = \epsilon_{min}$), we terminate the community seed set detection step. This exploration path ensures that first high quality communities will be discovered and subsequently less profound ones will also be detected. In order to speed up the parameter exploration process, we employ a logarithmic sampling strategy when moving along the μ parameter axis. The computational complexity of the proposed parameter scheme is a multiple of the original SCAN. The multiplicative factor is $C = s_\epsilon \cdot s_\mu$, where s_ϵ is the number of samples along the ϵ axis ($\simeq 10$) and s_μ is the number of samples along the μ axis ($\simeq \log k_{max}$).

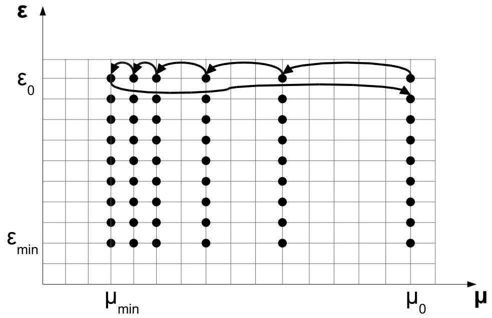

Fig. 2. Depiction of the (μ, ϵ) parameter space exploration path. The upper values μ_0 and ϵ_0 are set in relation to their maximum possible ones ($\mu_{max} = k_{max}$ and $\epsilon_{max} = 1.0$). The lower values are set to $\mu_{min} = 4$ and $\epsilon_{min} = 0.4$ since cores with lower values than these are of inconsistent quality.

3.3 Core Set Expansion

Starting from a community seed set S, the second step in the proposed community detection method involves an expansion process, which aims at attaching additional nodes, which are relevant, to the initial community seed set. The expansion step is essential for deriving higher quality communities since the community seed sets produced by the previous step may fail to include in the communities nodes that are of importance for them. In the case of tag communities, this would lead to tag communities that would miss some important keywords and would thus be less representative of their topic. In addition, it is due to this expansion step that overlap among communities is possible since the previous step produces non-overlapping community seed sets.

The community expansion step is based on the maximization of a local measure of community quality, namely *subgraph modularity* introduced in [16]. The modularity of a subgraph $S \in V$ is defined as the ratio of the number of intra-community edges (edges connecting nodes within S) over the number of edges sticking out of S (Equation 3). Obviously, the larger such a value is, the more well separated the subgraph is from the rest of the graph. In the extreme case of a disconnected subgraph, its modularity value tends to infinity:

$$M(S) = \frac{ind(S)}{outd(S)} = \frac{|\{(v,w) \in E | v, w \in S\}|}{|\{(v,w) \in E | v \in S \wedge w \in V - S\}|} \tag{3}$$

The proposed expansion step is based on a greedy maximization scheme, i.e. it successively attaches nodes to community S as long as their addition increases the subgraph modularity $M(S)$ of the community. The set of nodes that are considered as candidates for attachment to S are pooled from the "community frontier", i.e. the set of all nodes that are adjacent to at least one node of the community. Each candidate node is tentatively attached to the community and the new value of its modularity is computed. This computation can be performed very efficiently in an incremental fashion based on the values of $ind(S)$ and $outd(S)$ before the tentative attachment of the candidate node to the community.

Nodes with very high degree[4] are not considered in this process for two reasons: (a) to reduce the computational complexity of the expansion step, (b) to prevent the expansion process from creating a "gigantic" community. The node resulting in the maximum increase of modularity for the community is considered a member of the community and the process is repeated for the rest of the candidate nodes (it is possible that there is no increase of modularity by adding a node to the community, in which case no expansion takes place).

4 Evaluation

In order to gain insights into the behavior of community detection in real-world tagging systems, we conduct an evaluation study comparing the performance

[4] We create a degree-ordered list of nodes for the whole graph and consider as high-degree nodes the top 10% of them.

Table 1. Folksonomy datasets used for evaluation

(a) Basic folksonomy statistics

Dataset	#triplets	U	R	T
BIBSONOMY-200K	234,403	1,185	64,119	12,216
FLICKR-1M	927,473	5,463	123,585	27,969
DELICIOUS-7M	7,501,032	112,950	1,332,796	251,352

(b) Tag graph statistics (for large component)

Dataset	$\|V\|$	$\|E\|$	$\overline{k}$	$\overline{cc}$
BIBSONOMY-200K	11,949	236,791	39.63	0.6689
FLICKR-1M	27,521	693,412	50.39	0.8512
DELICIOUS-7M	216,844	3,443,367	31.76	0.8018

of our method (HGC) against two competing community detection methods on three datasets coming from different tagging applications, namely BibSonomy, Flickr and Delicious. The first of the two community detection methods under study is the well-known greedy modularity maximization scheme presented by Clauset, Newman and Moore (CNM) [18][5] and the second is the SCAN algorithm of [15], which is extended by HGC. The three datasets used for our study are described below and basic information on their size is presented in the upper part of Table 1.

BIBSONOMY-200K: BibSonomy is a social bookmarking and publication sharing application. The BibSonomy dataset was made available through the ECML PKDD Discovery Challenge 2009[6]. We used the "Post-Core" version of the dataset, which consists of a little more than 200,000 tag assignments (triplets) and hence the label "200K" was used to form the dataset name.

FLICKR-1M: Flickr is a popular online photo sharing and organizing application. For our experiments, we used a focused subset of Flickr comprising approximately 120,000 images that were located within the city of Barcelona (by use of a geo-query). In total, the number of tag assignments for this dataset approaches one million.

DELICIOUS-7M: Delicious is a popular social bookmarking service for managing and sharing bookmark collections. We used a snapshot of the Delicious bookmark collection corresponding to January 2006, comprising seven million tag assignments. This dataset is a subset of the collection studied in [19].

Starting from each dataset, we built a tag graph, considering an edge between any two tags that co-occur in the context of some resource. The raw graph contained a large component and several very small components and isolated nodes. For the experiments we used only the large component of each graph,

[5] We used the publicly available implementation of this algorithm, which we downloaded from `http://www.cs.unm.edu/~aaron/research/fastmodularity.htm`

[6] `http://www.kde.cs.uni-kassel.de/ws/dc09`

which accounts for more than 99% of the size of the raw graph for all three datasets. Some basic statistics of the analyzed large components are presented in the lower part of Table 1. The nodes of the three tag graphs appear to have a high clustering coefficient on average, which indicates the existence of community structure in them. We applied the three competing clustering schemes, CNM, SCAN and HGC, on the tag graphs and proceeded with the analysis of the derived communities. Since SCAN is parameter-dependent, we performed the clustering multiple times for many (μ, ϵ) combinations and selected the best solution.

Our first observation concerns the community structure produced by CNM. When considering the applications of tag clustering, it is hard to imagine that the highly imbalanced cluster structure produced by CNM can be of much benefit. For instance, knowing that two tags belong to the same huge cluster is not very informative of their semantic relation; in fact, there are many pairs of tags within such huge clusters that are not actually related to each other. Table 2 presents several such examples of unrelated tags which were placed in the same cluster. Having these tags in the same cluster is not only uninformative but it is actually misleading and thus potentially harmful for use within some IR task.

Table 2. Examples of unrelated tags that were assigned by CNM to the same community. Examples from the three largest communities of each dataset are presented.

Dataset	Examples of unrelated tags in the same community
BIBSONOMY-200K	hannover, nutritional, ebusiness, bishop, vivaldi, sunsets, skyscapes, recycle, antiracist, patentbibliometrics
	informationretrieval, magnetic, robotics, kolmogorov, wordnet, socialinformatics, thermodynamics, metaphysics, ...
	webdesign, windows, torrent, puzzle, vmware, geotagging, mov, techcrunch, cpplib, baseballplayers
FLICKR-1M	spanien, common chimpanzee, star wars, renault, restaurant, prostitution, olympicstadium, large windows, infrared
	barcelona, watermelon, photon awards, birthday, mediterranean, palm tree, fine arts, volkswagen, building, logistics
	roma, double bass, crowd surfing, environment, lomography, flickr babes, sombrero, basketball, bruce springsteen
DELICIOUS-7M	geekiness, telepathy, scifihorror, britneyspears, theflintstones, sportculture, environmentalhealth, uspatent, argentina, ...
	education, capetown, flashwebsites, businessanalyst, newjournalism, adventuretravel, musicnetwork, scienceastrophysics, ...
	food, island, bike, jersey, federal, climate, ghosts, athletics, enviroment, imperialism

In contrast, Table 3 presents several examples of interesting tag clusters discovered by HGC. Close examination of the tags contained in them reveals their close semantic and contextual association. In the case of CNM these clusters are contained in the aforementioned gigantic communities together with numerous unrelated tags, thus their utility is limited. On the other hand, the plain SCAN method can only identify subsets of these clusters, which is expected to harm the recall performance of the IR applications making use of them.

Table 3. Examples of interesting tag communities discovered by HGC. In the case of CNM, these communities are "hidden" within the gigantic communities discovered by CNM. In contrast, in the case of SCAN, these communities are smaller since they do not include tags from the community expansion step.

Dataset	**Examples of interesting HGC tag communities**
BIBSONOMY-200K	mpg, tif, jpeg, mpc, ico, wma, swf, fileconversion, txt, midi, psd, wmi, ogg, avi, psp, tiff, odg, mdb, kar, divx, wmv, qcp, odp, ods, rtf, odt, jpg, mov, amv, png, flv, flac, mmf, gif, sxw, amr, ...
	israelis, middleast, terrorism, middleastpeace, peaceprocess, onevoice, palestinians, conflictresolution, extremism, hatred
	urlogic, lymphatic, neoplasms, virus, pathophysiology, microbial, hemic, physician, doctor, musculoskeletal, respiratory, student, hepatological, viral, infections, hematological, gastrointestinal
FLICKR-1M	salad, spansih gastronomy, catalan food, modena, bacallà, colmenillas, bread with tomato, marinated, gastronomy, merluzzo, ec, marinado, cod, vinegar, bacalao, foie, meatfest, duck foie, ...
	george clooney, sean connery, jude law, antonio banderas, jennifer lopez, tom cruise, penelope cruz, viggo mortensen, ...
	series, australian, federer, conde godó, open, moya, tenerife, atp, las palmas gran, garros, torneo, murray, tamarasit, roland, roddick, podcast, bernardes, sharapova, djokovic, wta, wawrinka, campeonato, canarias, usopen, enric molina, chela gran, ...
DELICIOUS-7M	apollomission, saturnrocket, spacecrew, crewflight, navylieutenant, flightcommander, colonelwhite, americanastronauts, lieutenantcolonel, edwardwhite, spacewalk, capekennedy
	herbiehancock, dextergordon, chrispotter, brianblade, grantgreen, adamrogers, donaldbyrd, theloniousmonk, leemorgan, larrygoldings, hardbop, weatherreport, marcjohnson, mainstreamjazz, artblakey, billevans, joehenderson, joshuaredman, charlieparker, ...
	danacarvey, commercialparodies, thehanukkasong, richardpryor, stevemartin, wilferrell, chrisfarley, billmurray, adamsandler, kingtut, alecbaldwin, mikemyers, churchlady, chevychase, ...

Finally, we used the derived tag clusters in the context of tag recommendation in order to quantify their effect on the IR performance of a cluster-based tag recommendation system. More specifically, we created a simple recommendation scheme, which, based on an input tag, uses the most frequent tags of its containing cluster to form the recommendation set. In case more than one tags are provided as input, the system produces one tag recommendation list (ranked by tag frequency) for each tag and then aggregates the ranked list by summing the tag frequencies when of tags belonging to more than one list. Although this recommendation implementation is very simple, it is suitable for benchmarking the utility of cluster structure since it is directly based on it.

The evaluation process was conducted as follows: We divided the available tag assignments for each dataset into two sets, one used for training and the other used for testing. Based on the training set, we built the corresponding tag graph and produced the tag clusters based on the three competing methods. Then, by using the tag assignments of the test set, we quantified the extent to which the cluster structure found by use of the training set could help predict the tagging activities of users on the test set. For each test resource tagged with L tags, $K < L$ tags were used as input to the tag recommendation algorithm and the rest $L - K$ were predicted. In that way, both the number of correctly predicted tags and the one of missed tags is known. In addition, a filtering step was applied on the tag assignments of the test set. Out of the test tag assignments, we removed the tags that (a) did not appear in the training set, since it would be impossible to recommend them and (b) were among the top 5% of the most frequent tags, since in that case recommending trivial tags (i.e. the most frequent within the dataset) would be enough to achieve high performance.

Table 4 presents a comparison between the IR performance of tag recommendation when using the CNM, SCAN and HGC tag clusters. According to it, using the HGC tag clusters results in far better tag recommendations than by use of CNM across all three datasets. For instance, in the FLICKR-1M dataset, the HGC-based recommendation achieves six times higher precision than the CNM-based one (22.98% compared to 3.73%). A large part of the CNM-based recommendation failure can be attributed to the few gigantic communities that dominate its community structure. Compared to the best run of SCAN, HGC performs better in terms of number of unique correct suggestions, recall and P@1, but worse in terms of precision. In terms of F-measure, SCAN performs slightly better in two out of the three datasets, but HGC performs better in the third dataset. Given the fact that SCAN requires parameter tuning to achieve this performance and that HGC provides more correct unique suggestions, we conclude that the HGC tag cluster structure is more valuable in the context of tag recommendation. Since HGC extends SCAN in two steps (multiple iterations of SCAN and expansion of communities), we also ran tests to establish the relation of performance change to each of these steps: the multiple SCAN iteration step was responsible for a small part of the drop in precision and a measurable part of the increase in recall, while the expansion step was the main reason behind the increase in recall and the largest part of the drop in precision.

Table 4. IR performance of CNM, SCAN and HGC community structures in tag recommendation. The following notation is used: R_T denotes the number of correct tags according to the ground truth, R_{out} the number of tag suggestions made by the recommender, R_{TP} the number of correct suggestions, U_{TP} the number of unique correct suggestions, P, R, and F stand for precision, recall and F-measure respectively, and P@1, P@5 denote precision at one and five recommendations respectively.

	BIBSONOMY-200K			FLICKR-1M			DELICIOUS-7M		
	CNM	SCAN	HGC	CNM	SCAN	HGC	CNM	SCAN	HGC
R_T		15,216			55,875			56,893	
R_{out}	15,056	4,958	11,814	55,605	22,463	49,851	56,166	13,974	33,107
R_{TP}	272	1,120	**1,406**	2,074	10,419	**11,454**	1,022	3,624	**6,258**
U_{TP}	189	717	**837**	305	1,399	**1,666**	459	1,506	**2,628**
P (%)	1.81	**22.59**	11.90	3.73	**46.38**	22.98	1.82	**25,93**	18.90
R (%)	1.79	7.36	**9.24**	3.71	18.65	**20.50**	1.80	6.37	**11.00**
F (%)	1.80	**11.10**	10.40	3.72	**26.60**	21.67	1.81	10.23	**13.91**
P@1 (%)	1.68	3.96	**5.09**	1.95	8.02	**9.85**	1.64	2.78	**7.95**
P@5 (%)	2.18	**29.06**	17.27	3.41	**46.84**	21.27	2.35	**36.91**	29.49

5 Conclusions

We presented a parameter-free graph-based clustering scheme that is particularly suited to the task of tag clustering. The proposed scheme is based on the discovery of (μ, ϵ)-cores for multiple sets of (μ, ϵ) values and a subsequent expansion based on a local measure of cluster quality. We evaluated the proposed scheme on three real-world datasets and compared its performance against a modularity maximization clustering algorithm (CNM) and the basic (μ, ϵ)-core detection scheme (SCAN), which our proposal extends. We demonstrated that the tag clusters produced by our method are of significantly higher quality than the ones derived by CNM and achieve higher performance when used in the context of tag recommendation. Compared to SCAN, our method produces clusters with higher coverage (i.e. containing more related tags to the cluster topic). In the task of tag recommendation, the HGC clusters resulted in higher recall, but lower precision compared to SCAN. In addition, they led to a higher number of unique correct recommendations. Given also the fact that SCAN needs parameter tuning, we consider our clustering scheme as more suitable for identifying groups of related tags in folksonomies.

Acknowledgments. This work was supported by the WeKnowIt and GLOCAL projects, partially funded by the European Commission, under contract numbers FP7-215453 and FP7-248984 respectively.

References

1. Mathes, A.: Folksonomies - Cooperative Classification and Communication Through Shared Metadata (2004), `http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html`

2. Vander Wal, T.: Folksonomy Coinage and Definition (2007), http://www.vanderwal.net/folksonomy.html
3. Mika, P.: Ontologies are us: A unified model of social networks and semantics. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 522–536. Springer, Heidelberg (2005)
4. Hotho, A., Jäschke, R., Schmitz, C., Stumme, G.: Information Retrieval in Folksonomies: Search and Ranking. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011, pp. 411–426. Springer, Heidelberg (2006)
5. Begelman, G., Keller, P., Smadja, F.: Automated Tag Clustering: Improving search and exploration in the tag space (2006), http://www.pui.ch/phred/automated_tag_clustering
6. Simpson, E.: Clustering Tags in Enterprise and Web Folksonomies. Technical Report HPL-2008-18 (2008)
7. Au Yeung, C.M., Gibbins, N., Shadbolt, N.: Contextualising Tags in Collaborative Tagging Systems. In: Proceedings of 20th ACM Conference on Hypertext and Hypermedia, Turin, Italy, June 29-July 1, pp. 251–260. ACM, New York (2009)
8. Brooks, C.H., Montanez, N.: Improved annotation of the blogosphere via autotagging and hierarchical clustering. In: Proceedings of WWW 2006: 15th International Conference on World Wide Web, pp. 625–632. ACM, New York (2006)
9. Gemmell, J., Shepitsen, A., Mobasher, B., Burke, R.: Personalizing Navigation in Folksonomies Using Hierarchical Tag Clustering. In: Song, I.-Y., Eder, J., Nguyen, T.M. (eds.) DaWaK 2008. LNCS, vol. 5182, pp. 196–205. Springer, Heidelberg (2008)
10. Giannakidou, E., Koutsonikola, V.A., Vakali, A., Kompatsiaris, Y.: Co-Clustering Tags and Social Data Sources. In: Proceedings of WAIM 2008: 9th International Conference on Web-Age Information Management, pp. 317–324. IEEE, Los Alamitos (2008)
11. Java, A., Joshi, A., Finin, T.: Detecting Commmunities via Simultaneous Clustering of Graphs and Folksonomies. In: Proceedings of WebKDD 2008: KDD Workshop on Web Mining and Web Usage Analysis (2008)
12. Sigurbjörnsson, B., van Zwol, R.: Flickr tag recommendation based on collective knowledge. In: Proceedings of WWW 2008: 17th International Conference on World Wide Web, pp. 327–336. ACM, New York (2008)
13. Li, X., Snoek, C.G.M., Worring, M.: Learning Social Tag Relevance by Neighbor Voting. IEEE Transactions on Multimedia 11(7), 1310–1322 (2009)
14. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Physical Review E 69, 026113 (2004)
15. Xu, X., Yuruk, N., Feng, Z., Schweiger, T.A.: SCAN: A Structural Clustering Algorithm for Networks. In: Proceedings of KDD 2007: 13th International Conference on Knowledge Discovery and Data Mining, pp. 824–833. ACM, New York (2007)
16. Luo, F., Wang, J.Z., Promislow, E.: Exploring Local Community Structures in Large Networks. In: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 233–239. IEEE Computer Society, Los Alamitos (2006)
17. Papadopoulos, S., Kompatsiaris, Y., Vakali, A.: Leveraging Collective Intelligence through Community Detection in Tag Networks. In: Proceedings of CKCaR 2009 Workshop in K-CAP 2009 Conference, Redondo Beach, California, USA (2009)
18. Clauset, A., Newman, M.E.J., Moore, C.: Finding community structure in very large networks. Physical Review E 70, 066111 (2004)
19. Wetzker, R., Zimmermann, C., Bauckhage, C.: Analyzing social bookmarking systems: A del.icio.us cookbook. In: Proceedings of ECAI 2008 Workshop on Mining Social Data (MSoDa), Patras, Greece, pp. 26–30 (July 2008)