RATC: A Robust Automated Tag Clustering Technique[*]

Ludovico Boratto[1], Salvatore Carta[1], and Eloisa Vargiu[2]

[1] Dip.to di Matematica e Informatica, Università di Cagliari, Italy
boratto@sc.unica.it, salvatore@unica.it
[2] Dip.to di Ingegneria Elettrica ed Elettronica, Università di Cagliari, Italy
vargiu@diee.unica.it

Abstract. Nowadays, the most dominant and noteworthy web information sources are developed according to the collaborative-web paradigm, also known as Web 2.0. In particular, it represents a novel paradigm in the way users interact with the web. Users (also called prosumers) are no longer passive consumers of published content, but become involved, implicitly and explicitly, as they cooperate by providing their own resources in an "architecture of participation". In this scenario, collaborative tagging, i.e., the process of classifying shared resources by using keywords, becomes more and more popular. The main problem in such task is related to well-known linguistic phenomena, such as polysemy and synonymy, making effective content retrieval harder. In this paper, an approach that monitors users activity in a tagging system and dynamically quantifies associations among tags is presented. The associations are then used to create tags clusters. Experiments are performed comparing the proposed approach with a state-of-the-art tag clustering technique. Results –given in terms of classical precision and recall– show that the approach is quite effective in the presence of strongly related tags in a cluster.

1 Introduction

The developement of Web 2.0 applications, like blogs and wikis, led to a continuous growth of information sources, with daily uploaded resources shared by many users [1]. Besides traditional techniques to categorize and index data, new approaches based on collaborative tagging have been effectively proposed and adopted. The success of those approaches is due to the fact that tagging does not require specific skills and seems a natural way for people to classify any kind of resource.

A set of tags (tagspace) can be explored in several ways and many tagging systems usually define sets of related tags, called *tag clouds*, that help the tagspace visualization. However, as highlighted in [2], there are some well-known linguistic limitations that can inhibit information retrieval in those systems. In

[*] This is a draft version of the paper to appear in Proc. EC-Web 09, LNCS 5692, Springer-Verlang, 2009.

particular, the meaning or semantics of a tag is usually unknown. For instance, tag "orange" might refer either to a fruit or a color. Moreover, people use several tags to select the same resources. For example, a resource related to a pasta dish could be tagged as "Italian food", "spaghetti", "first course", etc. On the one hand, user can freely choose which tags classify resources in a useful way; on the other hand, the searching activity of other users within the tagspace could be limited. In fact, to find a resource it might be necessary to search several times using different keywords, and people should evaluate the relevance of the retrieved documents.

Grouping related tags together would avoid such limitations and simplify the exploration of a tagging system [3]. In fact, the definition of sets of related tags would help the identification of a context that would avoid polysemy and synononymy and make resources retrieval easier.

In this paper, we propose RATC (Robust Automated Tag Clustering), a technique that monitors users activity in the search engine of a tagging system in order to exploit implicit feedbacks provided by users. A feedback is collected each time a user finds a relevant resource during a search in a tagging system. The algorithm uses the feedback to dynamically strengthen associations between the resource indicated by the user and the tags used in the search string. Tag-resource associations are then used to infer tag-tag associations by adopting a standard correlation measure. Tag-tag associations allow to cluster tags in order to find strongly related tag sets. Results have been compared with the ones obtained by adopting the state-of-the-art approach proposed in [4] showing an improvement in the presence of strongly related tags in a cluster.

The main contribution of the proposed approach is that, by supervising users activity in a tagging system and monitoring their searches, we can progressively create and update tag-resource associations and tag-tag associations, rewarding the real semantic relations among tags and penalizing the misleading ones.

The rest of the paper is organized as follows: section 2 presents state-of-the-art in tag clustering; section 3 contains a detailed description of the steps we followed to build our technique; section 4 describes the performed experiments and outlines main results; section 5 discusses conclusions and future work.

2 Related Work

In the literature, several techniques, aimed at grouping tags by adopting different clustering algorithms and heuristics, have been presented.

In [5], an approach that tries to infer the semantics behind a tag space is proposed. The corresponding collaborative tagging can help in finding groups of concepts and partial ontologies. This is achieved by using a combination of shallow pre-processing strategies and statistical techniques together with knowledge provided by ontologies available on the semantic web. This technique starts pre-processing the data and cleaning up the tag space, then, evaluating co-occurrences, it finds tag-tag associations and clusters them. Semantic relations are extracted from the clusters and the results consist of groups of highly related tags that conceptualize specific facets of knowledge and correspond to elements in ontologies. This approach differs from the one proposed in this paper since our technique does not pre-process the tagspace. Our approach, in fact, is able to adaptively remove noisy tags by monitoring user interactions.

In [6], being aimed at extracting ontologies, authors proposed a way to integrate a social network with collaborative tagging. The usual tripartite models of ontologies based on users, tags and instances, are integrated with user-user relations. Concepts in each community (called *p-concepts*) are considered different and this model was used to resolve the polysemy/homonymy problem. This technique aims to group p-concepts and find keywords associations by using an algorithm that considers the interactions among users and p-concepts. Our approach differs in the sense that we consider users interaction just to link resources to tags, without creating explicit associations among users and resources.

In [7], an approach aimed at generating groups of semantically related tags through a probabilistic model is presented. The technique is based on evaluating co-occurrence of tags, resources, and users. The approach proposed in this paper differs because it does not rely on a probabilistic model and it does not consider users.

In [8], a co-clustering approach, based on the one proposed in [9], is employed. In this approach, tags and resources belonging to different datasets are clustered together. The clustering activity is based on a similarity metric that uses tag co-occurrences and semantic knowledge about the tags. The relations among the elements are used to enrich ontologies and to train multimedia processing algorithms. On the contrary, in our approach, the clustering activity is based just on tags and new knowledge is inferred by clustering elements of the same dataset.

In [10], a technique to exploit information from queries is presented. Associations between the keywords used in a query and the relevant resources retrieved by a search engine are exploited in order to rank search results based on the past users activity. The technique proposed in this paper creates associations also between a resource and the tags used to classify it when uploaded.

In [11], an approach to create clusters of queries is presented. Related queries are clustered together, in order to recommend a better query to users. This is achieved by finding the most descriptive words in a cluster and recommending better queries to users. In our approach, queries are used in a different way. We do not infer associations among tags clustering queries themselves, but tags associations are derived considering the resources that they classify.

In [4], a technique to cluster strongly related tags is presented. The algorithm is based on counting the number of co-occurrences (tags that are used for the same resource) of any pair of tags and a cut-off point is determined to decide if the co-occurrence count is significant enough to be used. Tags are clustered with an algorithm that is based on the spectral bisection and uses the modularity function to measure the quality of a partitioning. Related tags are then automatically discovered by incrementing a counter for each pair of tags that belong to the same cluster. Although the approach presented in this paper is quite similar, the main difference is that tag-resource associations are continuously updated during the use of the system.

3 RATC: Robust Automated Tag Clustering

RATC, which stands for Robust Automated Tag Clustering, monitors users activity in the search engine of a tagging system. The technique has been defined "robust" to put into evidence its ability to overwhelm the misleading resource classification problem.

3.1 Top Level View of the Approach

RATC encompasses four main steps:

***Tag-resource* associations creation.** As in any tagging system, each time a new resource is put into the system, a *tag-resource* association is created among that resource and the tags used to describe it.

Dynamic *tag-resource* associations evaluation. Users activity in the tagging system search engine is monitored and exploited in order to update existing *tag-resource* associations and to create new ones.

***Tag-tag* associations creation and quantification** Dynamic *tag-resource* associations are exploited to create associations among tags (*tag-tag* associations). A standard cosine similarity measure [12] is used to evaluate the similarity among tags. The result of this process is a weighted graph (*tag similarity graph*) in which each node represents a tag and each weighted arc represents the similarity value of the tags it connects.

Clustering. The community detection algorithm proposed by [13] is applied to the tag similarity graph in order to identify the intrinsic communities of tags.

3.2 Representation of a Tagging System

A tagging system is a community driven tool that allows users to classify resources by using tags. It can be represented as a bipartite graph that contains:

- a set T of tags t;
- a set R of resources r;
- a set $A : (T \times R)$ of weighted arcs $t - r$, representing tag-resource associations. The weight of the tag-resource associations represents the number of times that a tag has been associated to a resource by users.

As depicted in Fig.1, a tagging system is composed by a set of tags (rectangular nodes) linked by a weighted arc to a subset of resources (round nodes). In the example in figure there are three resources concerning with "goal actions" in a soccer game [3]. All of those resources has been classified with the tags *soccer* and *goal* and the weight of each arc represents the strength of the association

[3] Resources represent multimedia documents, like videos or pictures.

between a tag and a resource. Each tag has some outgoing dotted arcs, which indicate that there are other resources linked to those tags, not depicted in the example.

As a final remark, let us note that our approach does not take into account different meaning associated to a same word (i.e. polysemy). For instance, in the example in Fig.1, tag *goal* is used either as a successful attempt at scoring in a soccer game or as the place designed at the end of a race.

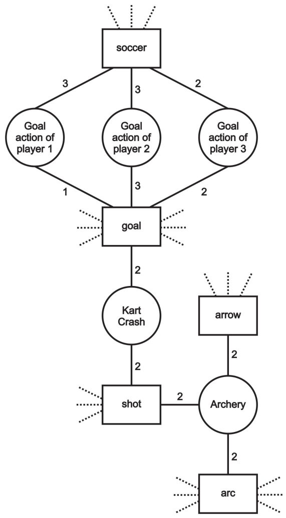

Fig. 1. A tagging system example

3.3 Tag-Resource Associations Quantification

The standard search paradigm provided by tagging services is based on query strings containing one (or more) tag. The search returns a list of resources associated to these tags. To provide such list, a ranking of the results is derived according to the tag-resource associations available in the tagging system that can be considered as the strength of the association between a resource and each tag used to describe it. While tagging systems usually associate tags and resources at upload time, implicit user feedback, coming from its search activity, can be exploited to improve *tag-resource* associations.

To represent the strength of *tag-resource* associations we adopted an algorithm based on counters. The algorithm exploits users feedback to discover and emphasize correct associations strength, while making negligible the contribution of "noisy" associations. The strength of each association evolves according to an extremely simple and effective mechanism. A *tag-resource* association is created each time a resource is added to the system by a user. After a search operation based on a tag, each time the user selects a resource, the counter of the tag-resource association is increased (an example of tag-resource associations is shown in Fig.1). Although a huge number of resources may be related to a single tag, their relevance will depend on the feedbacks provided by the community of users. In such a way the association of a misleading tag to a resource will give a negligible contribution.

In order to contain the counters relative to tag-resource associations, a matrix $W = \{w_{rt}\}$ is defined, where w_{rt} is the association between resource r and tag t (an example is depicted in Fig.2).

Initial values are assigned when a new resource is uploaded and values are updated either when a user adds a tag already present in the database or when a feedback is given [4]. When a new resource is uploaded to the tagging system together with some tags, the corresponding *tag-resource* counter is set to 1. If such association is already present in the system, the corresponding w_{ij} is incremented. The matrix is also updated when a user performs a search in the tagging system and selects one of the results as relevant. At this stage, after the user selection took place, the counters between the selected resource and all the tags in the query list are incremented, namely $w_{rt} = w_{rt} + 1$.

The tagging system shown in Fig.1 has been built using the tag-resource counters described above. Let us stress the fact that the strength of the relation between a tag and a resource in our tagging system is based on the feedbacks left by the users during the use of the system. For example, tag *soccer* has been used three times to classify and search the second resource concerning with the goal action of a player.

3.4 Tag-Tag Associations Quantification

Let v_i be the vector of associations among a tag i and its related resources and v_j be the vector of associations among a tag j and its releated resources. The association a_{ij} between tag i and tag j can be measured by the cosine similarity between the vectors as follows:

$$a_{ij} = cos(v_i, v_j) = \frac{v_i \cdot v_j}{\|v_i\|_2 \times \|v_j\|_2} = \frac{v_{i1} \cdot v_{j1} + ... + v_{ik} \cdot v_{jk}}{\|v_i\|_2 \times \|v_j\|_2}$$

These associations can be represented in a graph, call *tag similarity graph*, which links each couple of associated tags with a weighted arc. An example, built using the associations among tags and the resources shown in Fig.1, is represented in Fig.2 [5].

[4] The initial values are just a starting point that evolve as the feedbacks are collected.

[5] The values of the associations in the figure have been calculated considering the whole tagging system.

	Goal action of player 1	Goal action of player 2	Goal action of player 3	Kart Crash	Archery
soccer	3	3	2	0	0
goal	1	3	2	2	0
shot	0	0	0	2	2
arrow	0	0	0	0	2
arc	0	0	0	0	2

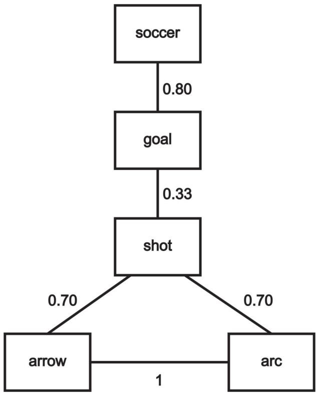

Fig. 2. Similarities graph

3.5 Clustering

To perform clustering we adopted MCL (Markov Clustering Algorithm) [13]. MCL is a Community Detection [14] algorithm, built to find cluster structure in simple graphs, considering the similarity between vertexes. The MCL algorithm tries to simulate the flow within a graph, considering just the part where the flow is strong and removing weak connections. If natural groups are in the graph, the links between the groups disappear, leaving the cluster structure.

The flow is simulated by a transformation of the graph into a Markov graph (a graph where for all nodes the sum of the weights of outgoing arcs is 1) and is expanded by computing powers of the associated stochastic (Markov) matrix.

Since these operations are not enough and do not reveal the clusters in the graph, a new operator (inflation) is inserted. Flow inflation [13] is the entry-wise HadamardSchur product of the matrix combined with a diagonal scaling, while flow expansion is represented by the usual matrix product. The inflation operator has been introduced to strengthen and weaken the flow, and the expansion operator is responsible for allowing flow to connect different regions of the graph.

As the MCL algorithm basically consists of alternation of two different operators on matrices, followed by interpretation of the resulting limit, its formulation is quite simple. It is also possible to find clusters of different granularities, by varying its parameters.

A more detailed description of this algorithm is beyond the scope of this article. The interested reader could refer to [13] for further details.

4 Experiments and Results

To evaluate the proposed approach, first, we adopted a tagging system with an internal experimental search engine [15], and then, we compared the performances with a state-of-the-art approach [4].

Several aspects have been taken into account while performing comparisons regarding the robustness of the two approaches. In particular, to analyze the impact of noise in the performances, we suitably added noisy tags to the tagging system. The quality of the obtained clusters have been evaluated comparing results with the ones provided by a domain engineer in terms of precision and recall. The adaptive capability of our approach (i.e., the users activity monitoring and their feedback) has been evaluated measuring the temporal evolution of clusters quality.

4.1 Setting Up the Experiments

To conduct the experiments 10 volunteers populated a tagging system [15] (*resource acquisition* step). They were asked to select as many videos as they wanted from YouTube [6] and add them to the tagging system. The application domain was limited to "sport" as specific topic, which can be considered a concept domain. Each video was classified with four tags related to the resource and two tags (the noisy tags) not related to the resource. Noisy tags were added to simulate the noise that typically occurs in practice.

Once the tagging systems was populated, volunteers were asked to perform normal searches in the tagging system (*feedback collection* step). During this step, RATC improves its performances monitoring users search activity. Videos are chosen based on a preview shown to the user and their original description. This step started as soon as the resource acquisition step was completed. The reason was to neatly separate the initial values of the correlations from their evolutions caused by the feedbacks of the users.

Resources Acquisition Each time a volunteer added a new video to the application, she/he had to create two sets of tags. The former is devoted to contain (at least) four characteristic tags, strongly related to the video; the latter is devoted to contain two tags not related to the video but in the same domain (in this experiments, "sport"). This tag set is required to create some verifiable noise and it has been used to monitor the progressive decreasing of their correlation with the video they had been initially introduced with. Such noise is useful to evalute the clustering algorithm. In particular, in this way we are able to monitor how the clusters structure changes and to evaluate the quality of the clusters.

The tagging system was populated with a total of 406 videos, 1021 tags, 2597 video-tag correlations. Although in [2] authors show that tags that identify qualities or characteristics can be effective in recommendation systems, being interested in clustering tags according to their meaning, we disregarded such kind of tags (i.e., those that express emotions or feelings). At the end of this step, the system involves 964 tags.

Feedback Collection During this step, each volunteer performed 300 searches in the tagging systems. For each search, each volunteer: (i) entered a list of tags

[6] http://www.youtube.com

as query for the search; and (ii) selected, from the videos in the results list, the video most related with the query.

A feedback is then collected each time a user performed a search and consequently tag-resource counters are incremented. After entering a list of tags, she/he was free to analyze the videos resulting from the search (during this phase the user could also play all the videos to help her/his choice). At the end of this activity, the user had to pick a video from the output list providing a feedback. This emulates a real world scenario in which a user, after the result of a search is displayed, selects the resources she/he is interested in.

4.2 Benchmark algorithm description

The technique selected for comparison with RATC, i.e., the one proposed by [4], hereinafter ATC.

ATC is aimed at clustering tags to improve user experience in the use of a tagging system and minimize the classical linguistic limitations. The approach defines an algorithm to find strongly related tags counting the number of tag co-occurrences used for a page. A cut-off point is determined to evaluate when a counter is useful. A spars matrix is produced and its elements are the similarities among tags.

A graph representation of the similarities is defined and the tags are grouped with a graph clustering algorithm based on the spectral bisection. The quality of the partitioning is measured with the "modularity function" Q [16]. ATC performs the following steps: (i) it uses spectral bisection to split the graph into two clusters; (ii) it compares the value of the modularity function Q_0 of the original unpartitioned graph to the value of the modularity function Q_1 of the partitioned graph, if $Q_1 > Q_0$ accepts the partitioning, otherwise rejects the partitioning; and (iii) it proceeds recursively on each accepted partition.

A similarity counter is increased for each pair of tags that belong to the same cluster and the top similar pairs of tags are extracted.

The choice to compare RATC with this approach is motivated by the fact that both approaches use tag-resource counters to define associations among tags. Let us also note that the main difference is on the way the counter is incremented. In fact, as previously explained, RATC counter is incremented also during the search activity.

4.3 Evaluation Measures

To assess the ability of RATC to learn from users activity monitoring, the state of the tagging system (i.e. the current values of each tag-resource association) has been saved and used to evaluate clusters quality each 50 feedbacks. In this way, 6 tagging system sessions, which can be used to compare the two tag clustering approaches, are available. As already pointed out, a subset of known tags was added to the tagging system to create some verifiable noise. To evaluate the quality of the clusters created by each algorithm in presence of noise we conducted experiments considering both the original dataset and a dataset in which we removed the noisy tags. The only parameter that had to be set is the inflation value in the clustering step (set to 3.0).

To make fair comparisons, first, a domain engineer clustered the involved tags. Each cluster was created considering tags that refer to the same *concept*, i.e. a particular event or a clear topic that groups tags. Subsequently, the tag clustering obtained by the domain engineer is compared with the clusters automatically generated by using RATC and the ones obtained by applying ATC. Each cluster produced by both RATC and ATC was evaluated considering the most related cluster generated by the domain engineer and producing the following sets:

- *true positive tags* (TP): tags that appear both in a cluster generated by RATC (ATC) and in the cluster of the domain engineer partition.
- *true negative tags* (TN): tags that do not appear both in a cluster generated by RATC (ATC) and in the cluster of the domain engineer partition.
- *false positive tags* (FP): tags that appear in a cluster generated by RATC (ATC) and do not appear in the cluster of the domain engineer partition.
- *false negative tags* (FN): tags that do not appear in a cluster generated by RATC (ATC), but appear in the cluster of the domain engineer partition.

To validate the approach, we resort to classical information retrieval measures, such as micro- and macro-averaging of precision and recall [17]. Let us recall here that micro- and macro-averaging are aimed at obtaining estimates of π and ρ relative to the whole category set. In particular, micro-averaging evaluates the overall π and ρ by globally summing over all individual decisions. In symbols:

$$\pi^{\mu} = \frac{TP}{TP + FP}; \quad \rho^{\mu} = \frac{TP}{TP + FN} = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} (TP_i + FN_i)} \tag{1}$$

where the "μ" superscript stands for microaveraging. On the other hand, macro-averaging first evaluates π and ρ "locally" for each category, and then "globally" by averaging over the results of the different categories. In symbols:

$$\pi^{M} = \frac{\sum_{i=1} mP_i}{m}; \quad rho^{M} = \frac{\sum_{i=1} mP_i}{m} \tag{2}$$

where the "M" superscript stands for macroaveraging.

4.4 Results

Fig. 3 compares the results in terms of macro-averaging precision (Fig. 3-a) and recall (Fig. 3-b) obtained by adopting RATC and ATC with and without noisy tags. Fig. 4 compares the results in terms of micro-averaging precision (Fig. 4-a) and recall (Fig. 4-b) obtained by adopting RATC and ATC with and without noisy tags. Results show that RATC performs always better than ATC, and that such performances improve session by session, due to the fact that tag-resource

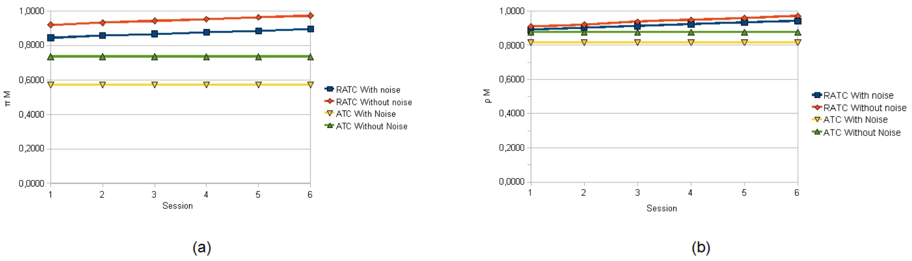

Fig. 3. Macro-averaging precision (a) and recall (b)

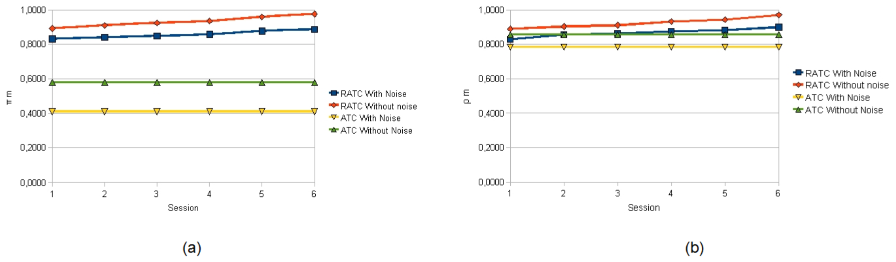

Fig. 4. Micro-averaging precision (a) and recall (b)

associations and tag-tag associations get better with the use of the system (i.e., by applying the feedback mechanism).

Let us put into evidence that, in the first session, the tag-resource associations have the same values for both algorithms, as no search activity was done in the system. Considering that RATC achieves better results even in this session, we can state that cosine similarity represents a better way to measure associations among tags.

5 Conclusions and Future Work

In this paper we proposed a technique able to cluster tags in a tagging system, with the ability to dynamically improve its performances while the tagging system is being used. The algorithm monitors users activity and exploits implicit feedbacks left by users. Experimental results highlight the effectiveness of the approach in the presence of strongly related tags in a cluster.

As for future work, we are currently studying how to improve the proposed approach by adopting a multi-layered clustering algorithm that, for each tag, takes into account the different contexts in which a tag is used.

Acknowledgments

We would like to thank Andrea Alimonda, Stefano Cossu, Luisella Piredda, and G. Michele Pinna for participating in the definition and implementation of the proposed system.

References

1. O'Reilly, T.: What is web 2.0? design patterns and business models for the next generation of software. www.oreilly.com (2005)
2. Golder, S.A., Huberman, B.A.: Usage patterns of collaborative tagging systems. J. Inf. Sci. **32** (2006) 198–208
3. Bielenberg, K., Zacher, M.: Groups in social software: Utilizing tagging to integrate individual contexts for social navigation. (2005)
4. Begelman, G., Keller, P., Smadja, F.: Automated tag clustering: Improving search and exploration in the tag space. (2006)
5. Specia, L., Motta, E.: Integrating folksonomies with the semantic web. (2007) 624–639
6. Hamasaki, M., Matsuo, Y., Nishimura, T., Takeda, H.: Ontology extraction by collaborative tagging with social networking. (In: WWW2008, April, Beijing, China)
7. Wu, X., Zhang, L., Yu, Y.: Exploring social annotations for the semantic web. In: WWW '06: Proceedings of the 15th international conference on World Wide Web, New York, NY, USA, ACM (2006) 417–426
8. Giannakidou, E., Koutsonikola, V., Vakali, A., Kompatsiaris, Y.: Co-clustering tags and social data sources. (2008) 317–324
9. Dhillon, I.S.: Co-clustering documents and words using bipartite spectral graph partitioning. In: KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, ACM (2001) 269–274
10. Smyth, B., Freyne, J., Coyle, M., Briggs, P., Balfe, E.: I-SPY: Anonymous, Community-Based Personalization by Collaborative Web Search. In: Proceedings of the 23rd SGAI International Conference on Innovative Techniques, Springer (2003) 367–380 Cambridge, UK.
11. Baeza-Yates, R.: Applications of Web Query Mining. In: Advances in Information Retrieval. Springer Verlag (2005) 7–22
12. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley (1999)
13. van Dongen, S.: Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht (2000)
14. Porter, M.A., Onnela, J.P., Mucha, P.J.: Communities in networks. (2009)
15. Carta, S., Alimonda, A., Clemente, M., Agelli, M.: Glue: Improving tag-based contents retrieval exploiting implicit user feedback. In Hoenkamp, E., de Cock, M., Hoste, V., eds.: Proceedings Of The 8th Dutch-Belgian Information Retrieval Workshop (DIR 2008). (2008) 29–35
16. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks (2003)
17. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys (CSUR) **34** (2002) 1–55