The Second International Workshop on Mining Communities and People Recommenders

Proceedings of the Workshop

COMMPER 2012

Editors:
Jaakko Hollmén
Panagiotis Papapetrou
Luiz Augusto Pizzato

ECML/PKDD 2012
September 28, 2012 – Bristol, UK

The Second International Workshop on Mining Communities and People Recommenders (COMMPER 2012)
`http://research.ics.aalto.fi/events/commper2012`

Sponsors:

Published: November 26, 2012

Preface

We are proud to present the workshop proceedings of The Second International Workshop on Mining Communities and People Recommenders (COMMPER 2012). The workshop was organized in conjunction with The European Conference on Machine Learning and Practice of Knowledge Discovery in Databases (ECML PKDD) in Bristol, UK on the 28th of September 2012. This workshop is a sequel to the first COMMPER Workshop, which took place in conjunction with the The IEEE International Conference on Data Mining series in Vancouver, Canada in December 2011.

The call for papers for COMMPER2012 was issued in the spring and distributed to e-mail lists for wide distribution. All submissions were reviewed by at least two members of the Program Committee. Finally, six contributions were accepted for inclusion in the program and the proceedings.

We wish to thank Evimaria Terzi, who kindly gave an invited talk in the workshop. and our sponsors for the financial support:

- Pascal2 – the European Network of Excellence in Pattern Analysis, Statistical Modeling and Computational Learning
- HIIT – the Helsinki Insitute for Information Technology
- the Smart Services Cooperative Research Centres (CRC)

Jaakko Hollmén, Panagiotis Papapetrou, Luiz Augusto Pizzato
Program Co-Chairs

COMMPER 2012 Committees

Workshop Co-Chairs

- Jaakko Hollmn, Aalto University, Finland
- Panagiotis Papapetrou, Birkbeck, University of London, UK
- Luiz Augusto Pizzato, University of Sydney, Australia

Program Committee

- Shlomo Berkovsky, CSIRO, Australia
- Aristides Gionis, Yahoo! Research, Spain
- Dimitrios Gunopulos, University of Athens, Greece
- Jaakko Hollmn, Aalto University, Finland
- Irena Koprinska, University of Sydney, Australia
- Theodoros Lappas, Boston University, USA
- Radhakrishnan Nagarajan, University of Arkansas, USA
- Panagiotis Papapetrou, Aalto University, Finland
- Irma Pasanen, Aalto University, Finland
- Luiz Augusto Pizzato, University of Sydney, Australia
- Antti Ukkonen, Yahoo! Research, Spain

COMMPER 2012 Program

Friday, September 28, 2012

10:30-10:45	Welcome Address
10:45-11:45	**Keynote:** Evimaria Terzi *Entity Selection and Ranking for Data-mining Applications*
11:45-12:15	**Introduction:** Challenges in the Area of Community Mining and People Recommenders

Session 1 - Mining Network Dynamics

12:15-12:45	Sofus Macskassy *Mining Dynamic Networks: The Importance of Pre-processing on Downstream Analytics*
12:45-13:15	Rushed Kanawati (presented by Manisha Pujari) Mining the Dynamics of Scientific Publication Networks for Collaboration Recommendation

13:15-14:45	Lunch

Session 2 - Mining Topics, Communities, and Reccommenders

14:45-15:15	Francesco Buccafurri, Gianluca Lax, Biagio Liberto, Antonino Nocera and Domenico Ursino *Supporting Community Mining and People Recommendations in a Social Internetworking Scenario*
15:15-15:45	Michael Meisel, Stefan Dahms and Andreas Ittner *Testing People To People Recommender in a Live Environment*
15:45-16:15	Derek Greene, Derek O'Callaghan and Padraig Cunningham *Identifying Topical Twitter Communities via User List Aggregation*
16:15 - 16:45	Nikolay Anokhin, James Lanagan and Julien Velcin *Social Citation: Finding Roles in Social Networks. An Analysis of TV-Series Web Forums*

16:45-17:30	**Discussion:** Challenges in the Area of Community Mining and People Recommenders
17:30-18:00	Concluding Remarks

Contents

Entity selection and ranking for data-mining applications

Evimaria Terzi

Boston University, Computer Science Department, Boston, MA, USA
evimaria@cs.bu.edu, http://www.cs.bu.edu/~evimaria/

Abstract. In many data-mining applications, the input consists of a collection of entities (e.g., reviews about a product, experts that declare certain skills, network nodes or edges) and the goal is to identify a subset of important entities (e.g., useful reviews, competent experts, influential nodes respectively). Existing work identifies important entities either by entity ranking or by entity selection. Entity-ranking methods associate a score with every entity. The main drawback of these approaches is that they ignore the redundancy between the highly scored entities. Entity-selection methods try to overcome this drawback by evaluating the goodness of a group of entities collectively. These methods identify the best set of entities, implying that all entities not in the group are unimportant. Such dichotomy of entities conceals the fact that there may be other subsets of entities with equally-good (or almost as good) goodness scores.

In this talk, we will discuss how the drawbacks of the above methods can be overcome by integrating the entity-ranking and entity-selection paradigms. That is, we will introduce entity-ranking mechanisms that are based on entity selection and entity-selection mechanisms that are based on entity ranking. In this framework, the importance scores of individual entities are determined by how many good groups of entities they participate in. Consequently, a good group of entities consists of entities with high importance scores. The main challenge we will discuss is how to explore the solution space of combinatorial problems in order to identify many entities that participate in many good solutions. In the talk, we will describe how our methods can be applied to applications related to expert management systems, management of online product reviews, and network analysis (including physical and social networks).

Mining Dynamic Networks: The Importance of Pre-processing on Downstream Analytics

Sofus A. Macskassy

Information Sciences Institute University of Southern California
Marina del Rey
CA 90292, USA sofmac@isi.edu,
WWW home page: http://www.isi.edu/~sofmac

Abstract. Dynamic networks are becoming ubiquitous and the analysis of these is becoming increasingly important to better understand the processes by which they evolve over time. Various recent work in this space have looked at how to detect communities, how to model adding/removing nodes and edges, and how to model how nodes change roles over time. In this paper, we look at a different aspect of dynamic networks: assuming we can identify and track a community over time, can we predict larger evolutionary events such as whether it is about to merge with another community or split or dissolve. Further, can we predict at a node-level whether nodes are about to leave the community. We here provide an initial exploration of how different settings for network extraction from observed time-stamped links impact community detection and performance of machine learning. We show on two data sets that changing these parameters have drastic impact on the consistency and stability of communities found.

Keywords: dynamic network analysis, social network analysis, community detection, machine learning, visual analytics

1 Motivation

The amount of social network data available for study is growing exponentially both in terms of complexity as well as in volume. Of particular interest is that this data is temporal in nature, enabling us unprecedented insight into how these networks evolve over time. One aspect of social networks which has received a lot of attention over the past decades is that of detecting communities and this problem translates directly into the dynamic networks as well where one can now detect and track communities over time in larger dynamic networks. There are various aspects of tracking communities which are of interest, but most work has focused on aspects of how communities evolve (e.g., [1, 7, 8, 5]). Most of the work in this area use community detection algorithms in some form to detect communities (e.g., [5, 11]). However, few have looked at the problem of how one actually generates the network from the dynamic data and what impact this has on performance of downstream analytics. We study in this paper how one can extract a "current" network from dynamic data, how this affects community detection and how this impacts machine learning on two prediction tasks: predicting whether a community is stable or about to merge with another community, and whether a node is likely to stay in a community or is about to leave.

To better understand the interplay between these aspects of network generation and analysis, we break up the problem into three parts:

1. Generating snapshots of what the "current" network looks like. We explore in this paper aggregating edges observed within a period of time and decaying edges from prior snapshots. While this generalizes to realtime updating as edges arrive as the period of time decreases to 0, we here explore larger aggregation periods.
2. Identify and track communities across snapshots. In this case, we use a standard modularity clustering algorithm (see [10]) to identify distinct communities in a snapshot and then track communities across snapshots using heuristics to decide whether two communities are the same based on membership overlap. This is in line with what others have done (see, e.g., [8]).
3. Evaluate how well machine learning can predict changes in nodes and communities as we change the parameters for taking snapshots. In particular, we explore whether we can predict if a community is about to merge or not or whether a node is about to leave a community. As such, this is a relatively simple prediction task, but turns out not to be that easy.

Each of these parts contribute to our understanding of dynamic network analysis, and the main contribution of the paper is that we show how the parameters we chose to take snapshots is quite important to downstream analytics. As we use this kind of paradigm to analyze dynamic networks, we need to take care in how these parameters are chosen.

We empirically show how changing the parameters for generating snapshots have drastic impact on the communities found and their stability over time. While this finding is not surprising, it shows the need to pay careful attention to the parameters and how they must be chosen both for the data set in question and for the goal of the analysis.

The remainder of the paper first describes the three parts of our network analysis (Sections 2-4). We then describe our data sets in Section 5 and describe our study of the impact of changing parameters in Section 6. We provide a short discussion of related work in Section 7 and finish with a discussion of our findings in Section 8.

2 Generating Network Snapshots

Dynamic networks as we define them in this paper are networks where edges are observed over time and these edges induces a network over the source and destination nodes. As such, an edge can be defined as e_{ij}^t, where the edge means that some relationship between node i and node j was observed at time t. This relationship can in general directed or undirected, have some semantic meaning and possibly a strength associated with it as well. In this paper we consider the edge to be generic and undirected though it can have a strength associated with it.

Assuming that most edges come at distinct time steps (e.g., there is only one edge observed at time $t \pm \epsilon$), then there would be no active "network" at time t. Therefore, in order to generate a network at time t, we need to consider observations into the past to create an aggregate network. However, an edge observed ten years ago is, in many cases, not as relevant as an edge observed one minute ago and so there is a question of how to deal with past observations.

The approach we take in this paper is to parameterize the network generation with three parameters: δ, the window of time which we consider to be "current" where all edges are taken as is; η, how much to decay edges observed prior to the current window, and γ, the threshold below which an edge is just not strong enough to be relevant and can therefore be removed. This threshold is important for computational reasons as the

network can otherwise become very dense and will have an impact on how well (and how fast) we can detect communities.

Using these three parameters, we can compute the network at time t. We can conceptually represent the network as an adjacency matrix A where the non-negative value $a_{ij} = a_{ji}$ represents the edge between node i and node j. If this is 0 then there is no relation between the nodes. Otherwise the value represents the strength of the relation. Using our three parameters, δ, α and γ, we compute A^t as follows:

$$A^t = A_0^t + \alpha * A^{(t-\delta)},$$

where $A_0^t = \{e_{ij}^{t'} | (t-\delta) \leq t' \leq t\}$. In other words, A_0^t is the network (adjacency matrix) induced by considering all "current" edges at time t. This is a recursive definition, going back to the start of our observed edges. We generate the final network G^t from A^t by pruning out edges whose value is less than γ:

$$\begin{aligned} G^t &= (V^t, E^t) \\ E^t &= \{a_{ij}^t | a_{ij}^t \geq \gamma\}, a_{ij}^t \in A^t \\ V^t &= \{v_i | i \in E^t\} \end{aligned}$$

To simplify notation and without loss of generality, we normalize time such that $\delta = 1$. Thus, given a set of edges over time, $E = \{e_{ij}^{t_0}, \ldots, e_{kl}^{(t_n)}\}$, we generate a set of snapshots: $G^1, \ldots, G^T$, where $G^1 = G^{t_0+\delta}$ and $G^T = G^{t_n}$.

3 Tracking Communities

Given a series of network snapshots, $G^1, \ldots, G^T$, we can now consider the question of identifying and tracking communities over time. We here assume that the dynamic network consists of multiple communities interacting over time and we want to track how these communities change from one snapshot to the next.

To do so, we must first be able to detect a community in a snapshot and second, we must be able to identify whether this community was present in a previous snapshot. While there are numerous community detection algorithms available (see, e.g., [10, 13, 8, 5, 9, 11]). We will in this paper use modularity clustering [10] to induce a set of disjoint communities $C^t = \{c_1^t, \ldots, c_k^t\}$ from G^t, where $c_i^t = \{v_j | v_j \in V^t\}, c_i^t \subseteq V^t$.

Given that we have $C^1, \ldots, C^T$, derived from $G^1, \ldots, G^T$ respectively, we need to identify whether c_i^t is a continuation of $c_j^{(t-1)}$. In fact, as has been enumerated elsewhere (see, e.g., [8, 5]), communities my take a set of actions between time-steps. We here define four major events which we track:

1. **Continue**: $\geq 50\%$ of $c_i^{(t-1)}$ moves on to c_j^t, and $\geq 65\%$ of c_j^t comes from $c_i^{(t-1)}$.
2. **Merge**: $\geq 50\%$ of $c_i^{(t-1)}$ moves on to c_j^t, but makes up $< 65\%$ of c_j^t.
3. **Split**: Significant portions ($> 30\%$) of $c_i^{(t-1)}$ moves on to multiple new communities (whether they join or create new communities is ignored).
4. **Death**: In this case, none of the above happened.

Using the above heuristics, we assign an action to each community at time t. From these, we categorize actions of nodes as well:

1. **Stay**: The community continues or merges and the node stays with that community.
2. **Leave**: The community continues or merges but the node goes to another community or does not belong to any community.
3. **Other:** The community splits or dies. Ignored in the study below.

4 Predicting Changes

The goal of this work was to develop predictive models to predict the action of a community or node at time t. In this paper, we looked at whether we could predict is a community would continue or merge and whether a node would leave or stay. All other actions were ignored for this exploration.

We framed this problem as a basic classification problem and used standard machine learning techniques to learn predictive models. The key factor here is how we generated attributes and labels. Clearly we can generate the labels (actions) automatically based on the processes described above, leaving us with how to generate attributes and the experimental methodology.

For communities, we computed a large set of metrics for each community at each time step. These included density, the ratio of closed triangles (triads) completely within the community or shared with another community (one or two of the nodes were in another community), the sum of edge weights within the community and going outside the community, and the closeness centrality of the most central people.

For nodes, we computed various structural attributes such as degree, ties within the community and to outside community, number of triads within the community and shared with another community, and closeness and betweenness centralities.

While we explored using trends for communities that lasted more than one timestep, we here only discuss results from using the metrics from a given snapshot.

For the experimental methodology, we varied α and γ and evaluated how well machine learning could predict the actions of communities and nodes in the induced networks. Our evaluation metric was area under the ROC curve. We performed 5x2 cross-validation [2], where we sub-sampled the majority class in the training set to make the two classes even because there was a very large class skew. However, we kept the test set at the large class skew. We test a variety of machine learning algorithms, including decision trees, logistic regression and naive Bayes. We used the Weka [14] machine learning toolkit. For brevity, we will only report results from logistic regression.

5 Data sets

We make use of two well-known and quite different data sets to show how changing parameters can have significant effect on downstream analysis.

The first data set is the **Enron email corpus** [6][1]. This data set contains the full email trace for roughly 150 enron employees spanning a period of three years (May 1999 through June 2002). The corpus contains over $500,000$ emails. For our purpose, we extracted the emails that were from one of the Enron employees to another employee. This left us with $60,409$ emails, containing a total of $139,183$ links as emails could have multiple recipients (in the to, cc or bcc fields). We used a δ of one month, meaning that we extracted one snapshot per month. The weight of a edge e_{ij} in a month is the number of emails between nodes i and j in that month.

[1] Available at `http://www.cs.cmu.edu/~enron/`

The second data set is the **World trade flow** data [4][2]. This data contains trade flow data between all countries from 1962 through 2000. While the data contains detailed statistics, we here use the overall trade information. This data details the full amount of imports and exports between two countries in a given year. Because trades increase over time, we normalized values such that the weight of edge e_{ij} between countries i and j denoted the ratio of all goods exported from i which went to j. Further, because this was a fairly dense network, we only kept the top 5 outgoing nodes from each country. We here used a δ of one year because that was the granularity of the particular data itself. The data contains information of 203 countries, not all of which were active in the beginning. The complete data (from 1962 to 2000) contained a total of $32,348$ links.

6 Study

The core of our study was focused on understanding the impact of changing α and γ both on the dynamics of the communities as well as on the performance of our classifier. We analyzed the two data sets with $\alpha \in \{0.5, 0.75, 0.9, 1.0\}$ and $\gamma \in \{0.05, 5.00\}$.

We first visualized how communities changed over time. Figure 1 shows one pair of visual analytics for the Enron data set. In order to keep the visual simple, we removed many of smaller communities as well as many of the edges showing how many people were moving between communities. However, the pair is still quite striking in that the analysis with lower α and higher γ has significantly more deaths and merges, and significantly fewer splits. This was a trend was saw on both data sets.

Second, we explored how well we could learn whether nodes were leaving or staying and whether communities would continue or merge. These were chosen because those were the most prominent classes. While we tested a variety of classifiers, we here report results from logistic regression.

	$\alpha = 0.50$			$\alpha = 0.75$			$\alpha = 0.90$			$\alpha = 1.00$		
	C	M	AUC	C	M	AUC	C	M	AUC	C	M	AUC
$\gamma = 0.05$	151	50	0.425	174	19	0.688	179	20	0.590	198	15	0.596
$\gamma = 5.00$	122	53	0.597	175	34	0.649	184	24	0.492	194	14	0.642

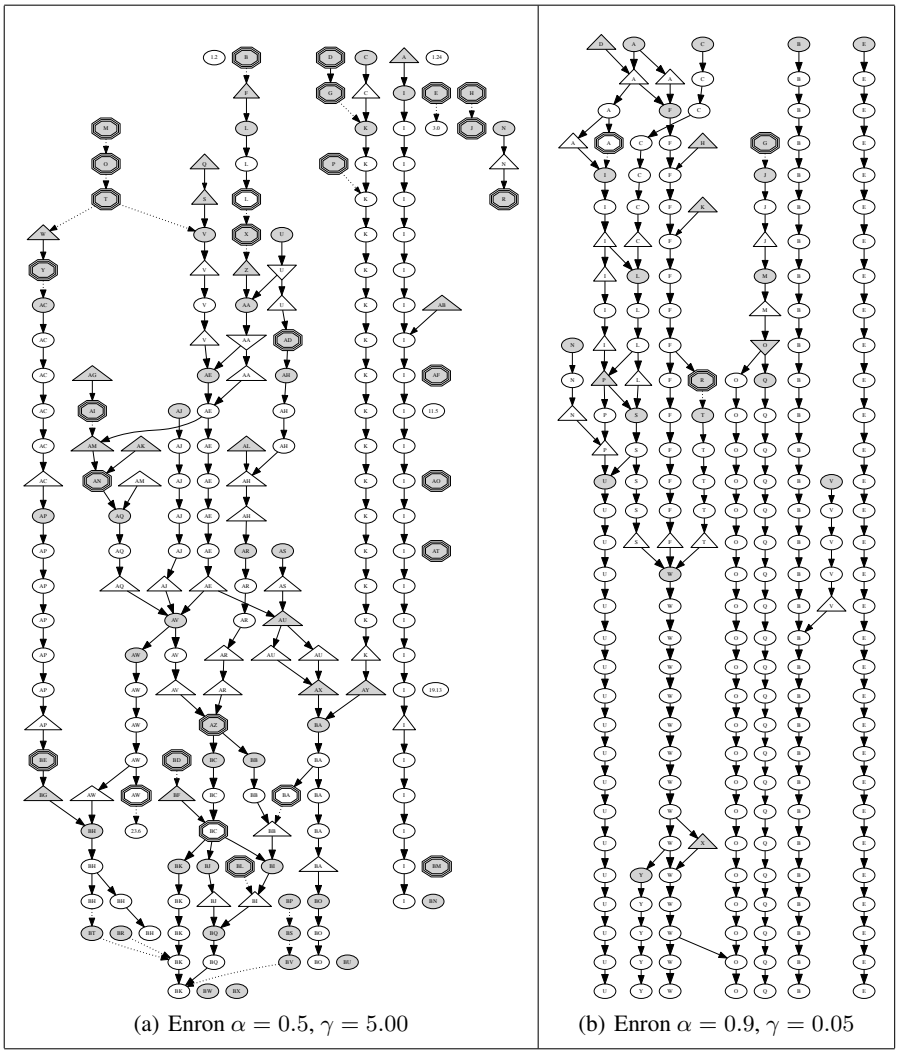
Table 1. Class skew and performance of logistic regression as we varied α and γ to predict community actions (C=continue, M=merge) on the Enron data set. As we can see, there was a strong effect of α on the class skew, but less of an effect with γ.

	$\alpha = 0.50$			$\alpha = 0.75$			$\alpha = 0.90$			$\alpha = 1.00$		
	S	L	AUC	S	L	AUC	S	L	AUC	S	L	AUC
$\gamma = 0.05$	3777	132	0.665	3847	107	0.704	3911	103	0.769	4001	0	1.000
$\gamma = 5.00$	2177	87	0.590	3230	59	0.682	3637	76	0.635	3729	78	0.744

Table 2. Class skew and performance of logistic regression as we varied α and γ to predict node actions (S=stay, L=leave) on the Enron data set. As we can see, both α and γ had a significant effect on class skew.

We first look at the Enron data set, where we varied α and γ to generate the evolving communities. We then used logistic regression to learn a classifier to predict actions for both communities and nodes. Table 1 shows the results of the generated communities

[2] Available at `http://www.nber.org/data/`



(a) Enron $\alpha = 0.5, \gamma = 5.00$ (b) Enron $\alpha = 0.9, \gamma = 0.05$

Fig. 1. Comparison of community evolution on the **Enron data** from 01/2000 through 06/2002 as we vary α and γ. Greyed out shapes mean that this was a new community. Circles means the community continued, triangles mean the community merged at the next step, inverted triangles meant the community split and octagons with thick borders meant the community died at that step. We see that with rapid decay (lower α) and higher γ, communities are much more volatile.

and learning the actions of these. As we can see, there was an effect of α but not much of an effect of γ on the class skew. Logistic regression did perform better than average as is shown by AUC values greater than 0.5 (except for two cases). However, performance did not correlate with α and γ. We see in Table 2 the same study when we learned to predict actions of nodes. In this case we see that both α and γ has significant impact on the class skew. We also see that these correlate well with the classifier performance.

	$\alpha = 0.50$			$\alpha = 0.75$			$\alpha = 0.90$			$\alpha = 1.00$		
	C	M	AUC	C	M	AUC	C	M	AUC	C	M	AUC
$\gamma = 0.05$	172	69	0.681	197	34	0.688	198	27	0.670	201	16	0.639

Table 3. Performance of predicting community actions on the world trade data communities. Again, we see strong impact of α on class skew, but little effect on classifier performance.

	$\alpha = 0.50$			$\alpha = 0.75$			$\alpha = 0.90$			$\alpha = 1.00$		
	S	L	AUC	S	L	AUC	S	L	AUC	S	L	AUC
$\gamma = 0.05$	6008	274	0.650	6233	246	0.691	6080	246	0.711	6484	181	0.660

Table 4. Performance of predicting community actions on the world trade data nodes. As before, we see strong impact of α on class skew, as well as some effect on classifier performance.

We next look at the World Trade Flows data. Tables 3 and 4 show the results of generating our clusters and predicting actions as we vary α and set $\gamma = 0.05$. As before, there is a strong impact on class skew and dynamics both for community and node actions (similar to what we saw in Figure 1). Although there was little change in performance for predicting actions of communities as shown in Table 3, we did see an impact of performance on predicting node activities as shown in Table 4. We are not quite sure why AUC dropped for $\alpha = 1.00$ and this is something that needs to be looked at closer.

7 Related Work

Of most relevance to this work are recent explorations of tracking the evolution of communities [8, 5]. Lin et al. [8] develop a framework for analyzing dynamic social networks, using generative models and stochastic block models. They use this model to track how a set number of communities interact over time, where the communities are identified from the aggregate graph. Greene et al. [5] look at the problem of tracking communities over time, identifying events such as birth, death, split and merge. They use a single similarity score to track communities from step to step and look at the volatility of community events as the threshold is changed.

The bulk of work in dynamic social networks has focused on various aspects of analyzing communities. For example Xu et al., have looked at evolutionary clustering [1] to identify optimal α at each time step to track clusters over time (see, e.g., [15]). This work focused on tracking clusters over time. This problem, and that of detecting communities in dynamic social networks, has also been looked at in various other published works (see, e.g., [13, 3, 9, 11]).

Most related to the prediction of nodes is the literature on predicting churn (nodes leaving the graph). While most work in this space uses non-network predictive models, Richter et al. [12] uses social and community features to identify potential churnes with some success.

8 Conclusion

Analyzing dynamic networks and communities is becoming more prominent both in the literature as well as in industry. However, dynamic networks also add complexity and we need to be mindful of the impact that data pre-processing has on downstream analytics. We have in this paper explored how changing the parameters of generating

snapshots of a dynamic network had a large impact on the observed dynamics of communities and nodes. We additionally saw that this impact on the dynamics also impacted performance of our classifiers when we learned models to predict actions of communities and nodes.

Acknowledgments

This work is based on research sponsored by the Air Force Research Laboratory (AFRL) under agreement number FA8750-12-2-0186. The views and conclusions herein do not represent those of AFRL or the U. S. Government.

References

1. Chakrabarti, D., Kumar, R., Tomkins, A.: Evolutionary clustering. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data mining (2006)
2. Dietterich, T.G.: Approximate statistical test for comparing supervised classification learning algorithms. Neural Computation 10(7), 1895–1923 (1998)
3. Duan, D., Li, Y., Jin, Y., Lu, Z.: Community mining on dynamic weighted directed graphs. In: Proceedings of the 1st ACM International workshop on Complex networks meet information and knowledge management (2009)
4. Feenstra, R.C., Lipsey, R.E., Deng, H., Ma, A.C., Mo, H.: World trade flows: 1962-2000. Working Paper 11040, National Bureau of Economic Research (January 2005), http://www.nber.org/papers/w11040
5. Greene, D., Doyle, D., Cunningham, P.: Tracking the evolution of communities in dynamic social networks. In: Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (2010)
6. Klimt, B., Yang, Y.: Introducing the enron corpus. In: Proceedings of the First Conference on Email and Anti-Spam (CEAS) (2004)
7. Leskovec, J., Backstrom, L., Kumar, R., Tomkins, A.: Microscopic evolution of social networks. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD) (2008)
8. Lin, Y.R., Chi, Y., Zhu, S., Sundaram, H., Tseng, B.L.: Analyzing communities and their evolutions in dynamic social networks. ACM Transactions on Knowledge Discovery from Data (TKDD) 3(2), 1–31 (Apr 2009)
9. McDaid, A., Hurley, N.: Detecting highly overlapping communities with model-based overlapping seed expansion. In: Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM) (2010)
10. Newman, M.: Modularity and community structure in networks. In: Procedings of the National Academy of Sciences. pp. 8577–8582 (2005)
11. Nguyen, N.P., Dinh, T.N., Xuan, Y., Thai, M.T.: Adaptive algorithms for detecting community structure in dynamic social networks. In: The 30th IEEE International Conference on Computer Communications (INFOCOM) (2011)
12. Richter, Y., Yom-Tov, E., Slonim, N.: Predicting customer churn in mobile networks through analysis of social groups. In: Proceedings of the SIAM International Conference on Data Mining. pp. 732–741 (2010)
13. Tantipathananandh, C., Berger-Wolf, T., Kempe, D.: A famework for community identification in dynamic social networks. In: Proceedings of the 13th ACM SIGKDD International conference on Knowledge Discovery and Data mining (2007)
14. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco (2000)
15. Xu, K.S., Kliger, M., III, A.O.H.: Tracking communities of spammers by evolutionary clustering. In: Proceedigns of the Workshop on Social Analytics: Learning from Human Interactions at the 27th International Conference on Machine Learning (2010)

Mining the dynamics of scientific publication networks for collaboration recommendation

Rushed Kanawati

LIPN, CNRS UMR 7030, Université de Paris13,
firstname.lastnam@lipn.univ-paris13.fr
F-93430 Villetaneuse, France

Abstract. In this paper we report on our experience on mining the dynamics of scientific publication networks for academic collaboration recommendation computation. We mainly focus on mining co-authorship networks. We show that dyadic topological link prediction approaches can be efficiently used for predicting the evolution off co-authorship networks. Different prediction approaches are introduced and evaluated including: supervised machine learning approaches and supervised rank aggregation techniques. We show, through extensive experimentation on real bibliographical data extracted from DBLP database, that mining the bipartite graph from which a co-authorship network is obtained (by simple projection on the author set) can significantly enhance prediction results.

1 Introduction

Online digital libraries play a major role in the overall process of the academic research. Beyond providing an easy and quick access to a huge amount of research outcome the availability of a sheer amount of computerized bibliographical data allow data miners to consider these digital repositories as an exciting target application field. A major trend in mining scientific publications consists on mining bibliographic networks that can be extracted from these databases [31]. A bibliographic network is simply a graph, in which vertices represent objects and links represent relations between objects. Main involved *objects* in a bibliographic network are *authors*, *papers* and *venues* (conferences as well as journals and books). Different types of relations link these objects (ex. write, cite, participate, etc.) defining a real complex heterogeneous network. Publications being dated, a temporal sequence of evolving networks can be readily obtained. However, most of existing work consider *static homogeneous* networks in which only one kind of objects exists linked by the same type of links. In addition, mostly all links are handled equally, regardless of the associated time-stamp. Examples of most studied homogeneous networks are :

- *Co-authorship networks* where nodes are authors. Two authors are linked if they co-sign at least one publication.
- *Citation networks* where nodes are papers. Each paper is linked to all other papers it cites.

– *Bibliographic coupling networks* where nodes are papers. Two papers are linked if they cite at least one common third paper.

Many other types of networks can be extracted including term-oriented networks such as term co-occurrence networks, and indexing networks if we consider Web 2.0 online bibliographical services such as Mendeley[1], CiteuLike[2] and Bibsonomy[3] to mention a few. Main targeted applications of mining bibliographic networks including efficient computing of bibliometrics indices, clustering and research community identification, identifying missing relevant citations [9].

In this work we are interested in mining the dynamics of bibliographic networks for academic collaboration recommendation. Computed recommendation can be used for recommending new contacts within the space of international conferences and meetings [25]. Another application would be to help in building up new research consortiums.

Recommendations are computed by applying a link prediction approach in a **heterogeneous** bibliographic network. The target link to predict is the co-authorship tie linking two authors. We are interested as others in predicting the formation of *new links* rather than repeated ones [22, 28]. Formally, the link prediction problem can be defined as follows: Let $< G_1, \ldots, G_t >$ be a temporal sequence of snapshots of a given network (i.e. graph). Our goal is to predict what *new* links in the graph G_{t+1} will appear between nodes belonging to a $G_{i,1 \leq i \leq t}$ but have never linked before.

The link prediction problem has attracted much of interests in the last few years. A variety of approaches have been proposed in the scientific literature. Recent surveys on the topic can be found in [24, 16]. A major trend is composed of topological approaches: these are approaches based merely on mining topological evolution of the network history in order to predict the appearance of new links [23]. Such approaches are inherently application-field independent. They spare the need for any specific knowledge about the actors (i.e. nodes) of the studied network. Meanwhile, these approaches can be combined with node content approaches for enhancing prediction performances [14].

Co-authorship networks have been frequently studied in the field of social network analysis [27]. Some earlier link prediction approaches have also been applied to this type of data [22, 28]. However, most of existing approaches consider only the homogeneous co-authorship network. In this work we report on our experience in studying the problem of co-authorship link prediction in a heterogeneous bipartite network linking authors to papers [5, 29]. We show through experimentation on real bibliographic networks extracted from the, now well known, DBLP[4] bibliographical server that mining the heterogeneous bipartite network enhance prediction performances. We follow, as others [15, 28, 26] a supervised link prediction approach where a set of training data is used to *learn*

[1] http://www.mendeley.com/
[2] http://www.citeulike.org/
[3] http://www.bibsonomy.org/
[4] http://www.informatik.uni-trier.de/ley/db/

a model for predicting new links. Two different supervised approaches are experimented: supervised machine learning and more originally supervised rank aggregation.

The contributions of this paper include:

- Studying the problem of co-authorship prediction in a heterogeneous bipartite bibliographic network.
- Comparing different supervised link prediction approaches, namely supervised machine learning and supervised rank aggregation on real bibliographic networks (DBLP).

The reminder of this paper is organized as follows. Next in section 2 we review briefly main related work studied in the state of the art. We mainly introduce the dyadic topological link prediction approaches and we show how these can be applied in a supervised link prediction framework. In section 3, we detail our proposed approaches. Results of experimentations are reported and discussed in section 4. Finally we conclude in section 5.

2 Related Work

2.1 Dyadic topological link prediction

The seminal work of *Liben-Nowell* and *Kleinberg* [23], has defined the basis of dyadic topological link prediction. The principle is to infer the future connectivity in a network by leveraging the current connectivity information. This is achieved as follows. Given the current state of the network, we compute for each indirectly connected couple of nodes (nodes are supposed to be in a connected sub-graph) a *topological feature* capturing some aspects of the relationship between both considered nodes. The list of unlinked couples of nodes is then sorted according to the value of this topological feature. Top k sorted elements are returned as predictions. The major problem here is to fix the value of k. Actually, rather then being a prediction approach this simple feature-based approach serves mainly to discover which topological features can efficiently inform link formation. In this context, k is set to the number of real new links appearing next in a testing period.

A wide variety of topological features can be computed. Before giving a brief account of attributes used in this study, we give here some basic notations that are applied afterwards. Let $\Gamma_G(x)$ be the set of direct neighbors of a node x in a graph G. This is denoted $\Gamma(x)$ when there is no ambiguity concerning the considered graph. $\parallel E \parallel$ is the cardinality of set E. The degree of a node x in a graph G is equal to $\parallel \Gamma_G(x) \parallel$. We denote by A_G the adjacency matrix of graph G. In all the following G is considered a homogeneous unimodal connected graph. Topological features can be grouped in three groups:

Combination of node's topological measures One well known effective link score measure is *preferential attachment (PA)* proposed in [3]. This is simply defined by the product of involved node's degrees:

$$PA(x,y) = \| \Gamma(x) \| \times \| \Gamma(y) \| \quad (1)$$

Building on this example, we an propose new topological features using other node-based topological measures. A long list of node-based topological metrics can be used such as: PageRank [7], the *hub* and the *authority* indicators computed by the HITS algorithm [21], the clustering coefficient, and the different centrality measures [32]. In this work we limit ourselves to the use of the preferential attachment and the product of page rank index:

$$PPR(x,y) = PR(x) \times PR(y) \quad (2)$$

Where $PR(x)$ is the page rank of node x.

Neighborhood based measures The second family of topological features are based on evaluating the overlap between neighborhoods of involved nodes. The most frequently applied measures in this context are the following: Common Neighbors (denoted $CN(x,y)$), Jaccard's coefficient (denoted $JC(x,y)$), and the Aadmic-Adar measure proposed in [1] (denoted $AA(x,y)$. These are respectively defined as follows:

$$CN(x,y) = \| \Gamma(x) \cap \Gamma(y) \| \quad (3)$$

$$JAC(x,y) = \frac{\| \Gamma(x) \cap \Gamma(y) \|}{\| \Gamma(x) \cup \Gamma(y) \|} \quad (4)$$

$$AD(x,y) = \Sigma_{z \in \Gamma(x) \cap \Gamma(y)} \frac{1}{log(|\Gamma(z)|}. \quad (5)$$

Distance-based measures A third class of link's score is based on evaluating paths linking involved nodes[5]. The simplest attribute to compute is the shortest path between x and y. We denote this attribute by $Dis(x,y)$. Another measure, frequently applied in affiliation network analysis, is *Katz coefficient* proposed in [20]. It consists in computing a weighted sum of all paths between x and y. More formally, the score of a link $< x, y >$ is given by:

$$Katz(x,y) = \Sigma_{l=1}^{\infty} \beta^{\ell} \times \| path_{x,y}^{(\ell)} \| \quad (6)$$

Where $path_{x,y}^{(\ell)}$ is the number of paths between x and y of length ℓ. β is a positive parameter which favors shortest paths. In [13], it is shown that the value of attribute $Katz$ for two nodes i and j is given by the element $K[i,j]$ where K is a matrix given by $K = (I - \beta \times A)^{-1} - I$, A is the adjacency matrix of the considered graph and I is the identity matrix. The computation of matrix K converges to the above formula if β is smaller than the inverse of the largest

[5] Recall that we compute examples for nodes belonging to a same connected component, hence at least one path exist between each couple of nodes

proper value of matrix A. Another similarity matrix between nodes of a graph is proposed in [8], where the similarity (indeed a distance function) is given by the matrix $T = (I + L)^{-1}$ where L is the laplacian metric of the considered graph. Other measures based on random walks through the considered graph [13] or exploiting the analogy with electrical circuits [12] are also proposed in the literature.

2.2 Supervised link prediction

Experiments conducted in [23] has shown that topological features defined earlier catch information about link formation in complex networks. However, prediction performances has shown to be poor. An obvious idea that has been quickly proposed in the scientific literature is to combine different topological features in order to learn a model explaining link formation. One early work is the one proposed in [15] where the problem of link prediction is formulated in terms of a binary classification problem allowing using supervised machine learning approaches. Generally, supervised link prediction can be described as follows. Let $G_{[t_i,t_j]}$ be the graph describing a network in time period $[t_i, t_j]$. We split the history of the network evolution into two time intervals :

- *Learning interval* $[t_0, t_1[$ is used to compute topological features characterizing couples of nodes that are not directly connected during this interval.
- *Labeling interval* $[t_1, t_2[$ is used to label couples of nodes identified in the precedent interval as `linking` or `not-linking`.

The topological feature vectors computed for each couple of unlinked nodes in the learning interval coupled with labels detected in the labeling interval define a classical binary supervised classification problem. These data are used to learn a model that can be validated on data generated in the same manner but by starting at time $t > t_0$. Different technics has been used in the literature to learn classification models including: supervised machine learning [15, 28, 26], semi-supervised machine learning [19] and logistic regression models [30].

In [17] authors adapt some of the above defined topological features to be applied to bipartite graphs. The goal is to predict links in a product purchase graph linking customers to products. The target link to predict is the one linking objects of different types. This is different from our case where we do consider bipartite graphs but where the goal is to predict a link in a projected graph (the co-authorship graph).

More related to our work is the work reported in [30] where authors propose also to mine heterogeneous bibliographic networks for co-authorship link prediction. In this work authors use a more complex heterogeneous graph including authors, papers, venues and topics. Data used are much richer than those used in our work. However the link prediction approach is different. It is based on learning to weight meta paths linking two authors passing by different types of objects (papers, venue and topics).

Our approach is based on introducing new topological features for qualifying couples of authors but taking into account their relations to other types of

objects, mainly publications. By using this new features, in addition to classical features computed in the homogeneous co-authorship network we can still use the simple propositionalisation approaches for learning link predictions models. In addition we propose a new supervised link prediction approach based on rank aggregation methods. These approaches are detailed in next section.

3 Proposed approaches

3.1 Topological features in bipartite network

In [5] we were the first to propose to mine a heterogeneous bibliographic network for co-authorship link prediction task. Instead of mining homogeneous co-authorship networks we propose to consider the original bipartite graph linking authors to published papers. A bipartite graph G is defined as follows: $G =< \top, \bot, E >$ where $\top$ and $\bot$ are two mutually exclusive sets of nodes and where endpoints of ties, composing the E set, come from different sets. A unimodal graph can be obtained from a bipartite one by projecting the graph over one of its node sets. For example, the projection over the $\top$ set is defined by a unimodal graph where nodes from $\top$ set are tied if they are linked to at least n common nodes in the initial bipartite graph G. In a more formal way, let $\Gamma_g(x)$ be the set of neighbors of node x in a graph g. Projections of a bipartite graph G are then defined as follows:

- $G^n_\top =< V_\top \subseteq \top, E = \{(x, y) : x, y \in \top, \parallel \Gamma_G(x) \cap \Gamma_G(y) \parallel \geq n\} >$
- $G^m_\bot =< V_\bot \subseteq \bot, E = \{(x, y) : x, y \in \bot, \parallel \Gamma_G(x) \cap \Gamma_G(y) \parallel \geq m\} >$

In order to take into account the original two-mode nature of the co-authorship network we propose in this work to compute for each example two sets of topological attributes:

- *Direct attributes.* These are simply topological measures defined for unimodal graphs. We compute these attributes in the co-authroship graph.
- *Indirect attributes.* We define an *indirect attribute* [5, 4] as an extension of a classical attribute (attributes shown above) which is calculated from the projected graph $G^m_\bot$ (i.e. the publication graph). These measures are based on the principle of propagation of similarities, it is used here to quantify the indirect similarities between *authors* by calculating the conventional similarities (or direct) between their *publications*. With each direct measure M (i.e. from the list given before) we can associate an indirect measure $M^m_\bot$ computed as follows:

$$M^m_\bot(x, y) = \Phi_{u \in \Gamma_{G_{bip}}(x), v \in \Gamma_{G_{bip}}(y)}(M(u, v))$$

  Φ is some aggregate function. In our study, the *max* aggregate function is applied, except for the shortest path attribute for wich *min* function is used.

We use these different types of topological features to learn a model using a supervised machine learning approach as described before. We show through experimentation reported in 4 that new indirect features enhance prediction performances in a significant way.

3.2 Supervised rank aggregation for link prediction

Another way to combine the predictability power of different topological features is to apply rank aggregation methods developed in the context of computational social choice theory [10]. Actually following the initial approach proposed in [23], each topological feature defines a rank on the list of unlinked couples of nodes. A straightforward way to combine the results is to apply classical rank aggregation methods [6, 11, 2]. One widely used (unsupervised) rank aggregation method is the one defined three centuries ago by Borde [6]. This functions as follows. A Borda score is calculated for each element in the lists. For a set of complete ranked lists $L = [L_1, L_2, L_3, ..., L_k]$, the Borda's score for an element i and a list L_k is given by:

$$B_{L_k}(i) = \{count(j)|L_k(j) < L_k(i)\&j \in L_k\} \tag{7}$$

The total Borda's score for an element is given as:

$$B(i) = \sum_{t=1}^{k} B_{L_t}(i) \tag{8}$$

The method consists simply on ranking elements by their decreasing total Borda score. A random decision is taken to rank sub-lists of elements having the same total Borda score. This method is simple to compute. However it is known not to be compliant with Condorcet principle stating that a winner of an election (i.e. the output of a rank aggregation process) should be the element that is preferred to each other element in the list by a majority of voters. In our case, voters are the topological features and elements are the couples of unlinked nodes to be evaluated. Computing a Condorcet compliant aggregation list provides better guarantees in case of noisy data than simple Borda aggregation. One approach for computing Condorcet-compliant rank aggregation is Kemeny aggregation [11]. This is based on computing a permutation of the element list that minimize the sum of the Kendall-Tau distance to all other lists. The Kendall-Tau distance counts the number of pairs of elements that have opposite rankings in the two input lists i.e. it calculates the pairwise disagreements:

$$K(L_1, L_2) =| \ (i, j) \ s.t. \ L_1(i) \leq L_2(j) \ \& \ L_1(j) \geq L_2(j) \ | \tag{9}$$

Instead of computing unsupervised rank aggregations, we propose to apply the supervised framework in order to learn weights to associate to each voters. These weights are introduced then in the rank aggregation process to compute links to be predicted.

We thus propose two ways to introduce weights into Borda's method and local Kemeny optimal method.

Supervised Borda: We introduce weights into Borda's method in the following way. Suppose $(w_1, w_2, \ldots, w_n)$ are the weights for n rankers (and thus for the ranked lists provided by them), then the Borda score for individual element can

be given by:

$$B(i) = \sum_{t=1}^{n} w_i * B_{L_t}(i) \tag{10}$$

Supervised Local Kemeny Aggregation:. Algorithm 1 describes our proposed approach for finding supervised local Kemeny aggregation. Details can be found [29].

Algorithm 1 Supervised local kemeny aggregation

Input: $T = [\tau_1, \tau_2, \ldots, \tau_r]$ where $\tau_i = [e_1, e_2, \ldots, e_m]$ for r rankers and m elements
$W = [w_1, w_2, \ldots, w_r]$ where w_i is the weight for ranker i and $w_T = \sum_{i=1}^{r} w_i$
$\mu = \tau_1$ where μ can be considered as initial aggregation
Output: π : an aggregated list of elements

Initialize an empty matrix M
for element $x = 1$ to $m - 1$ **do**
  for element $y = 1$ to m **do**
    $score = 0$
    for $\tau_i \in T$ **do**
      $$xPREFy = \begin{cases} 0 & \text{if } \tau_i(x) > \tau_i(y) \\ 1 & \text{if } \tau_i(x) < \tau_i(y) \end{cases}$$
      $score = score + (w_i * xPREFy)$
    end for
    if $score > 0.5 * w_T$ **then**
      $M_{xy} \Leftarrow true$
      $M_{yx} \Leftarrow false$
    else
      $M_{xy} \Leftarrow false$
      $M_{yx} \Leftarrow true$
    end if
  end for
end for
Bubble sort μ using M.
if $M_{xy} = false$ **then**
  swap(x,y)
end if
Return μ

Computation of weights: Weights of the topological features are computed based on the following criteria :

– **Maximization of positive precision:** Based on maximization of identification of positive examples the attribute weight is calculated as

$$W_{a_i} = n * Precision_{a_i} \tag{11}$$

where n is the total number of attributes and $Precision_{a_i}$ is the *precision* of attribute a_i based on identification of positive examples.

– **Minimization of false positive rate:** By minimizing the identification of negative examples we get a weight as below

$$W_{a_i} = \frac{n}{FPR_{a_i}} \tag{12}$$

where n is the total number of attributes and FPR_{a_i} is the $false\ positive\ rate$ of attribute a_i based on identification of negative examples examples.

4 Experiments

We evaluated our proposed approaches on heterogeneous bibliographic network extracted form the DBLP database. The data used corresponds to a time span of 1970 to 1980. This data is divided into three datasets containing information for different years each having a training set and a test or validation set. The training set is composed of a learning set spanning a period of 4 consecutive years and a labeling period spanning the following two years. For example the first dataset denoted $[1970-1973, 1974-1975]$ we use graphs in the period 1970 to 1973 to compute topological features. Labeling is done on period $[1974-1975]$. The validation dataset is then constructed by sliding the cursor one year further so using the dataset $[1971-1974, 1975-1976]$ The three generated datasets are described in table 1. The colon labels *positive* gives the number of links to predict while the colon *number* gives the total number of couples of nodes that are not connected in the training period. The table shows clearly the data skewness problem where the number of positive examples is very small compared to the number of negative examples. This makes the learning process challenging. However, handling the data skewness problem is out of the scope of this paper.

Datasets	**Training period**	**Validation period**	**Training examples**		**Test examples**	
			Positive	**Total**	**Positive**	**Total**
Dataset 1	[1970,1973,1974,1975]	[1971,1974,1975,1976]	30	1693	41	3471
Dataset 2	[1972,1975,1976,1977]	[1973,1976,1977,1978]	87	19332	82	18757
Dataset 3	[1974,1977,1978,1979]	[1975,1978,1979,1980]	102	35190	164	60046

Table 1. DBLP Datasets

The first experiment we have realized aimed at comparing the predicability power of direct and indirect topological features. We applied the same procedure proposed in [23]. Table 2 shows the average precision of each type of used topological features. It is clear that indirect features capture some information about the link formation which is different from this provided by direct attributes.

In a second experiment we searched to evaluate the contribution of the new indirect topological features on the model learning process. We compared the

Attributes	Dataset1	Dataset2	Dataset3
Katz	0	0	0.0061
MFA	0.0244	0.0732	0.0488
PPR	0.0244	0.0244	0
PCD	0	0	0
VC	0.5122	0.4268	0.1829
JC	0.2195	0.1707	0.0488
AD	0.1463	0.1463	0.1463
AP	0.0488	0	0
Dis	0	0.0122	0.0244
Indirect Dis	0.6098	0.5366	0.8171
Indirect AD	0.0976	0.0488	0.0366
Indirect AP	0.0244	0	0
Indirect PCD	0.0244	0.0122	0.0061
Indirect MFA	0.0488	0.0732	0.0427
Indirect JC	0.0488	0.1098	0.0549
Indirect PPR	0.0488	0	0
Indirect VC	0.0488	0.0488	0.0183
Indirect Katz	0.1220	0.1098	0.0488

Table 2. Results in terms of average precision obtained by ranking the test examples by attribute values

performances of predictions using a supervised machine learning approach (a boosted decision tree) using only direct attributes (those computed in the co-authorship network) with performance obtained from using the same learning method but processing examples using both direct and indirect attributes. Figure 1 shows clearly the positive contribution of indirect attributes.

Lastly, using all topological features, we have computed the performance of different proposed prediction approaches using:

- For supervised learning approach simple decisions tress and designs tress with boosting.
- Supervised rank aggregation using both Borda and Kemeny approaches. For both approaches we applied both proposed weighting schemes.

In order to deal with the problem of data skewness, or class imbalance we have randomly sampled the training data in order to have the number of negative examples as the double of the number of positive ones. The experiment is repeated 10 times and yes average results are reported on figure 2. Results are expressed in terms of F1-measure. F-measure is defined by the harmonic mean of both precision and recall.

$$F = 2 \times \frac{Precision * Recall}{Precision + Recall} \tag{13}$$

Figure-2 shows the results obtained on the complete datasets in terms of F1-measure.

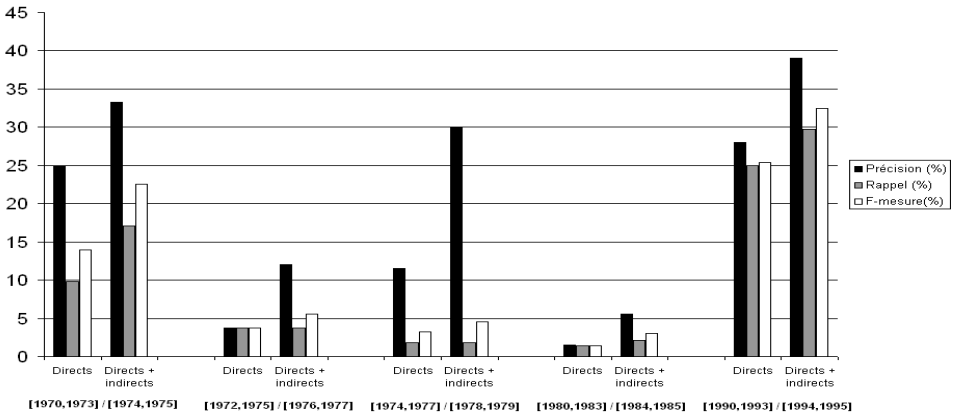


Fig. 1. Contribution of indirect attributes for learning link formation model

These results show the validity of proposed approach mainly the supervised rank aggregation approach. In figure 3 we compare the outcome of supervised rank aggregation with unsupervised rank aggregation approaches. Again results obtained show that the supervised version outperforms the basic unsupervised approach.

5 Conclusion

In this paper we have studied the problem of predicting co-authorship relationship in heterogeneous bibliographic network linking two types of objects: papers and authors. We have introduced new topological measures characterizing relation between authors that are computed taking into account the graph projected on the paper set. These new features has been proved to enhance the performances of learning approaches. We have also proposed a new supervised link prediction method based on the idea of supervised rank aggregation. Again results of experiments on real datasets show the validity of this approach.

We are working now on conducting experiments on more complex heterogeneous bibliographic networks involving more than two kinds of objects: namely including venues and topics of papers. Another research axe we are working one is to deal with high skewness of data and to deal with the problem of large scale of available data. We are examining the idea of restricting the application of the proposed link prediction approaches to a *community* level rather than considering the whole network. We've developed a new efficient algorithm for automatic community detection [18] and we are working on fine-tuning our link prediction approaches in order to learn link formation models inter and intra communities.

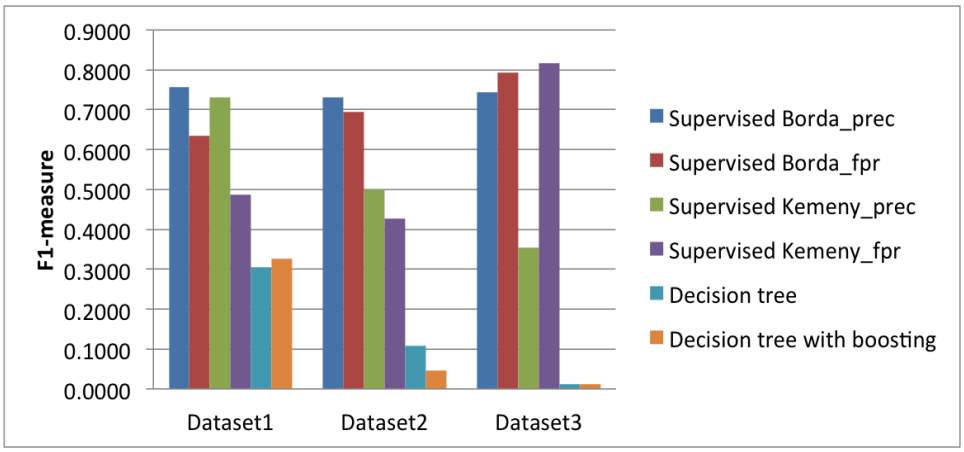


Fig. 2. Comparing different link prediction approaches using F1-measure

References

1. L. A. Adamic, O. Buyukkokten, and E. Adar. A social network caught in the Web. *First Monday*, 8(6), 2003.
2. L. Akritidis, D. Katsaros, and P. Bozanis. Effective rank aggregation for metasearching. *Journal of Systems and Software*, 84(1):130–143, 2011.
3. A. L. Barabasi, H. Jeong, Z. Neda, E. Ravasz, A. Schubert, and T. Vicsek. Evolution of the social network of scientific collaborations. *PHYSICA A*, 311:3, 2002.
4. N. Benchettara, R. Kanawati, and C. Rouveirol. Apprentissage supervisé pour la prédiction de nouveaux liens dans des réseaux sociaux bipartie. In *Actes da la 17iéme Rencontre de la société francophone de classification (SFC'2010)*, pages 63–66, St. Denis, La réunion, June 2010.
5. N. Benchettara, R. Kanawati, and C. Rouveirol. Supervised machine learning applied to link prediction in bipartite social networks. In *International Conference on Advances in Social Network Analysis and Mining, ASONAM 2010*, 2010.
6. J. C. Borda. Mémoire sur les élections au scrutin. *Comptes rendus de l'Académie des sciences, traduit par Alfred de Grazia comme Mathematical Derivation of a election system , Isis, vol 44, pp 42-51*, 1781.
7. S. Brin and L. Page. The anatomy of a large scale hypertextual web search. In *proceedings of seventh International conference on the world wide web*, 1998.
8. P. Chebotarev and E. Shamis. The matrix-Forest Theorem and measuring Relations in small Social Groups. *Automation and Remote Control*, 58(9):1505–1514, 1997.
9. D.-Z. Chen, M.-H. Huang, H.-C. Hsieh, and C.-P. Lin. Identifying missing relevant patent citation links by using bibliographic coupling in led illuminating technology. *J. Informetrics*, 5(3):400–412, 2011.
10. Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. A short introduction to computational social choice. *SOFSEM 2007: Theory and Practice of Computer Science*, pages 51–69, 2007.
11. C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *WWW*, pages 613–622, 2001.

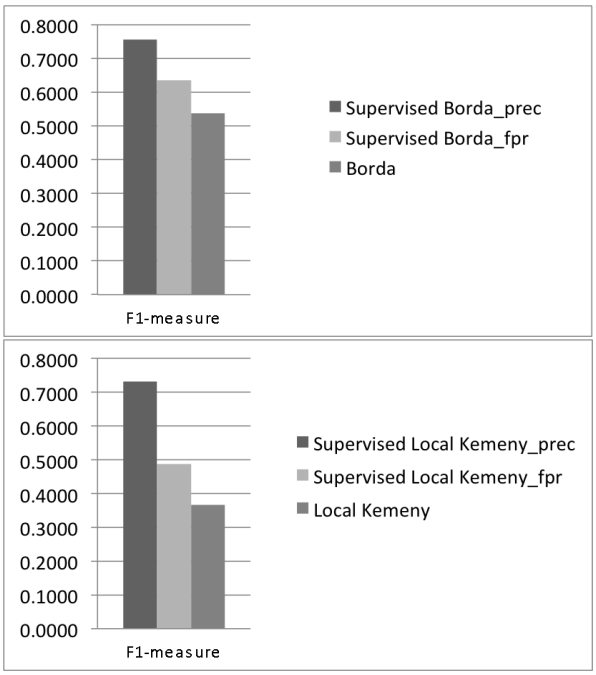


Fig. 3. Comparing supervised and unsupervised rank aggregation approaches

12. C. Faloutsos, K. McCurley, and A. Tomkins. A. Fast Discovery Of Connection Subgraphs. In *10th ACM Conference on Knowledge Discovery and Data Mining*, 2004.
13. F. Fouss, A. Pirotte, J.-M. Renders, and M. Sarens. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommandation. *IEEE Transactions on knowledge and data engineering*, 19(3):355–369, 2007.
14. S. Gao, L. Denoyer, and P. Gallinari. Temporal link prediction by integrating content and structure information. In C. Macdonald, I. Ounis, and I. Ruthven, editors, *CIKM*, pages 1169–1174. ACM, 2011.
15. M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link Prediction using Supervised Learning. In *SIAM Workshop on Link Analysis, Counterterrorism and Security with SIAM Data Mining Conference*, Bethesda, MD, 2006.
16. M. A. Hasan and M. J. Zaki. A survey of link prediction in social networks. In C. C. Aggarwal, editor, *Social Network Data Analytics*, pages 243–275. Springer, 2011.
17. Z. Huang, X. Li, and H. Chen. Link prediction approach to collaborative filtering. In M. Marlino, T. Sumner, and F. M. S. III, editors, *JCDL*, pages 141–142. ACM, 2005.
18. R. Kanawati. Licod: Leaders identification for community detection in complex networks. In *SocialCom/PASSAT*, pages 577–582. IEEE, 2011.

19. H. Kashima, T. Kato, Y. Yamanishi, M. Sugiyama, and K. Tsuda. Link propagation: A fast semi-supervised learning algorithm for link prediction. In *SDM*, pages 1099–1110. SIAM, 2009.
20. L. Katz. A new status index derived from socimetric analysis. *Psychmetrika, 18(1)*, 18(1):39–43, 1953.
21. J. Kleinberg. Authoritative sources in hyperlinked environments. *Journal of the ACM*, 4:604–632, 1999.
22. D. Liben-Nowell. *An Algorithmic Approach to Social networks.* PhD thesis, M.I.T., 2005.
23. D. Liben-Nowell and J. M. Kleinberg. The link-prediction problem for social networks. *JASIST*, 58(7):1019–1031, 2007.
24. L. Lu and T. Zhou. Link prediction in complex networks: A survey. *CoRR*, abs/1010.0725, 2010.
25. J. F. McCarthy, D. W. McDonald, S. Soroczak, D. H. Nguyen, and A. M. Rashid. Augmenting the social space of an academic conference. In J. D. Herbsleb and G. M. Olson, editors, *CSCW*, pages 39–48. ACM, 2004.
26. T. Murata and S. Moriyasu. Link Prediction based on Structural Properties of On-line Social Networks. *New Generation Computing*, 26:245–257, 2008.
27. M. E. J. Newman. Scientific collaboration networks. II. shortest paths, weighted networks, and centrality. *Phys. Rev. E*, 64(1):016132, 2001.
28. M. Pavlov and R. Ichise. Finding Experts by Link Prediction in Co-authorship Networks. In A. V. Zhdanova, L. J. B. Nixon, M. Mochol, and J. G. Breslin, editors, *FEWS*, volume 290 of *CEUR Workshop Proceedings*, pages 42–55. CEUR-WS.org, 2007.
29. M. Pujari and R. Kanawati. Supervised rank aggregation approach for link prediction in complex networks. In *International workshop on Mining Social Network Dynamics (MSND 2012)*, Lyon, April 2012. WWW'12.
30. Y. Sun, R. Barber, M. Gupta, C. C. Aggarwal, and J. Han. Co-author relationship prediction in heterogeneous bibliographic networks. In *ASONAM*, pages 121–128. IEEE Computer Society, 2011.
31. Y. Sun, T. Wu, Z. Yin, H. Cheng, J. Han, X. Yin, and P. Zhao. Bibnetminer: mining bibliographic information networks. In J. T.-L. Wang, editor, *SIGMOD Conference*, pages 1341–1344. ACM, 2008.
32. S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications (Structural Analysis in the Social Sciences).* Camridge University Press, 1994.

Supporting Community Mining and People Recommendations in a Social Internetworking Scenario

Francesco Buccafurri, Gianluca Lax, Biagio Liberto, Antonino Nocera and Domenico Ursino

DIMET, University Mediterranea of Reggio Calabria, Via Graziella, Località Feo di Vito, 89122 Reggio Calabria, Italy
{bucca,lax,b.liberto,a.nocera,ursino}@unirc.it

Abstract. Community mining and people recommendation are playing a key role in social networks, allowing a very large number of promising applications. Nowaday, Social Networking is evolving towards Social Internetworking, where the interaction among distinct social networks represents the main distinguishing feature. As a consequence, no effective results can be obtained if the analyst misses the heterogeneity of the current social-network universe together with the inter-relationships crossing different social networks and allowing the definition of a concept of community which is transversal over this universe. As a consequence, the preliminary step consisting in the dataset collection must necessarily take into account the features of this Social Internetworking Scenario (SIS, for short), which, instead, are not considered whenever a single social network is crawled. In this paper we present SNAKE, a system conceived to extract a large set of useful information in a SIS. It represents the basis of new-generation crawlers which can operate on a SIS capturing the intrinsic multi-context nature of the current social-network communities.

1 Introduction

In the last years, the enormous increase of the social network phenomenon led community mining [5] and people recommendation [14] to play a central role. Indeed, currently, community mining is extensively investigated and exploited for deriving virtual communities in social networks, for analyzing communities, for detecting human collaborations and social network models, for discovering the topics of main interests to people in social networks, and so forth. On the other hand, mining meaningful relationships among people joining social networks can facilitate the recommendations of people, which is an extremely challenging and specific task in this scenario. Most recently, people tend to join more social networks and, often, to provide different personal information in each of them. As a consequence, the interaction among distinct social networks is representing the basis of a new emergent Social Internetworking Scenario (SIS, for short) [13,9] enabling a lot of strategic applications whose main strength will be just

the integration of possibly different communities yet preserving their diversity and autonomy. Clearly, Social Internetworking Scenarios represent a challenging issue (and, at the same time, a challenging opportunity) also for community mining and people recommendations which should not miss this huge multi-network source of information that also reflects multiple aspects of people personal life [11] and that cannot be derived from the examination of a single social network. In other words, the internetworking phenomenon is changing various aspects of the social Web, including the concept of community itself, which takes new connotations of transversality over the constellation of social networks. As a consequence, no effective results can be obtained if the analyst misses the heterogeneity of the current social-network universe, together with the interrelationships crossing different social networks. This, in terms of actions to do in the preliminary phase consisting in the dataset collection, means that we must be able to extract data by strictly taking into account the features of this Social Internetworking Scenario. Unfortunately, this is not done by standard crawlers [10,7,12,8], designed for operating in a single social network. In fact, the design of a crawler for a SIS poses two specific problems. The former concerns the sampling strategy (for instance, how to obtain a sample taking data from the involved social networks and being representative of the considered SIS). The latter concerns the data extraction strategy (for instance, it regards the capability of operating with the different standards adopted by the involved networks to represent user relationships). Whereas the former problem has been already addressed [6], the latter one is still basically open.

In this paper, we deal with the second problem by proposing SNAKE (Social Network Account Knowledge Extractor), a system supporting data extraction in a multi-social-network scenario. SNAKE is able to return public information related to a social-network account, including those allowing the interconnection of different social networks, thus fully supporting crossing crawling of a SIS. In particular, the extracted information concerns user account details, user contacts and existing `me` edges. These last ones are edges linking accounts of the same user on different social networks. We point out that SNAKE acts as a middleware between social network data and *any kind* of crawler. A demo of SNAKE is available at the address `http://ictsud.unirc.it:8080/snake-demo.html`. We point out that the problem faced in this paper is basically open since, in the literature, only two commercial systems performing tasks someway similar to the ones handled by SNAKE were proposed, namely Google Social Graph (which was fully retired on April 20, 2012) [1], and Online Identity Consolidator [2]. However, there exist several, even strong, differences between them and SNAKE.

Google Social Graph (GSG, for short) aims at providing information about both relationships between different users of the same network and between different accounts of the same user on different social networks. It preliminarily extracts user data and stores them in a cache. This way, it can reduce the time necessary to answer user queries. However, this leads to the possibility that the answers are not updated and, therefore, fully reliable. By contrast, SNAKE does not exploit a cache mechanism and, therefore, can always provide updated

information. Anyway, in spite of this choice, the time necessary for it to answer user queries is satisfying, as shown in Section 3. A limitation of GSG concerns the dimension of query answers. In fact, GSG returns at most 10,000 contacts per user. Clearly, this limitation becomes quite valuable for the analysis of large social networks and can lead to errors. By contrast, SNAKE does not present any limitation on the number of contacts returned in its answers. Finally, GSG can handle only information represented by means of XFN [4] and FOAF [3]. By contrast, SNAKE is capable of handling not only these two formats but also Social Network APIs. As a consequence, it is capable of handling a higher number of social networks.

Online Identity Consolidator (OIC, for short) is a tool provided in Plaxo, a framework aiming at supporting address book and social network management services. There are also some differences between OIC and SNAKE. First, OIC is capable of handling only the XFN format. Furthermore, OIC returns only `me` edges in its answers, whereas SNAKE returns also information about the corresponding contacts. Finally, SNAKE is a multi-thread system.

This paper is organized as follows: in Section 2, we describe both the architecture of SNAKE and the corresponding components. In Section 3, we present the prototype implementing it, along with an experimental analysis of its performances. Finally, in Section 4, we draw our conclusions.

2 SNAKE Description

The general architecture of SNAKE is reported in Figure 1. It consists of five macro-areas, namely *SNAKE Front-end*, *API Management*, *FOAF Management*, *XFN Management* and *SN Front-end*.

SNAKE Front-end receives a URL representing the home page of a user account in a social network and constructs a social network node representing this account. *API Management* represents the system macro-area devoted to access and query the specific social network APIs. *FOAF Management*, instead, focuses on information about user contacts. *XFN Management* is in charge of extracting information about existing `me` edges through the XFN technology. Finally, *SN Front-end* encompasses all the resources provided by the social networks to extract available public information from their database. In the following we describe, in more details, the SNAKE macro-areas.

SNAKE Front-end. As illustrated in Figure 1, this macro-area consists of three modules, namely *SN Node Solver*, *SN Parser*, and *SN Model Handler*. The corresponding working flow is as follows. First, *SN Node Solver* is activated, which accepts the URL of the user account home page and analyzes it to create a query compliant with the social network which the account belongs to. After this, *SN Node Solver* activates *SN Parser* and passes the generated query to it. Then, *SN Parser* activates the *API*, *FOAF* and *XFN Management macro-areas* and passes them the query to execute. Finally, when these macro-areas have carried out their tasks, thus returning the expected information, *SN Parser* requires *SN*

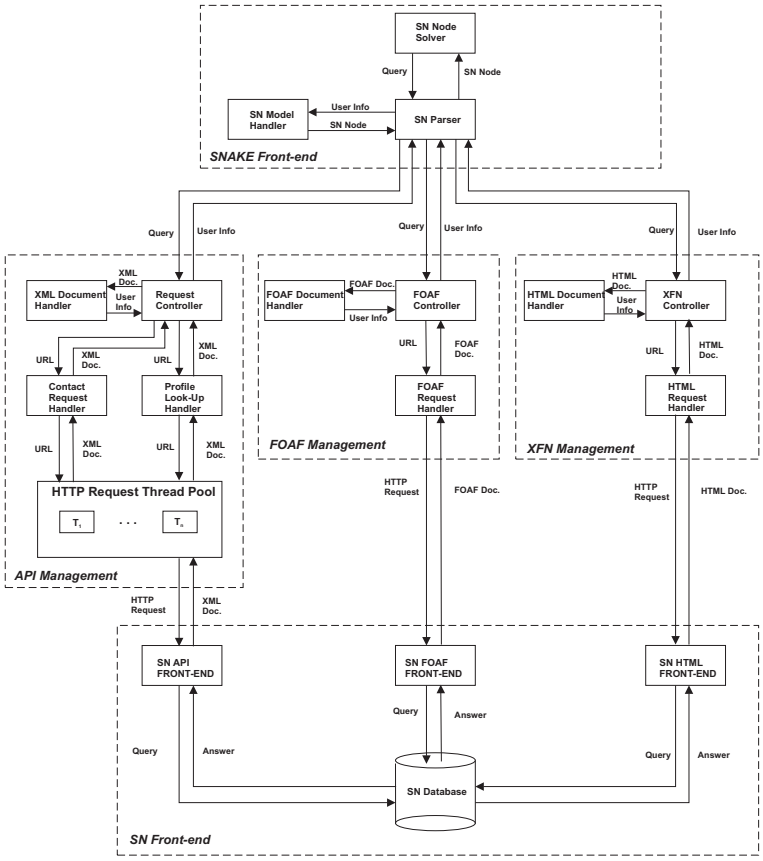


Fig. 1. Architecture of SNAKE

Model Handler to create a new SN node. When this node has been created, *SN Model Handler* passes it to *SN Parser*. This returns the SN Node to *SN Node Solver* which, in its turn, returns it to the caller.

API Management. As illustrated in Figure 1, this macro-area consists of five modules, namely *Request Controller*, *XML Document Handler*, *Profile Look-Up Handler*, *Contact Request Handler*, and *HTTP Request Thread Pool*. This macro-area focuses on the extraction of information about a user, her contacts and her `me` edges through a social network API. In particular, the information extracted by this macro-area regards the activities performed by a user inside the network, her contacts (along with the kind of relationship), her profile, her possible external links (e.g., `me` edges), etc. Clearly, SNAKE focuses on public data, i.e. data accessible without authentication.

As for the technicalities underlying APIs, the functionalities provided by their methods can be accessed by performing specific requests. These could be

encoded in different formats, such as `REST` (`RE`presentational `S`tate `T`ransfer), `XML-RPC`, `SOAP`, etc. The *API Management* macro-area adopts the most common request format, i.e. the one adopted in the `REST`-style architecture. This is based on the assumption that each resource can be addressed by a URI and can be represented by a document in the XML or the JSON formats. A resource request can be performed via the HTTP protocol by specifying a URL containing the resource URI and the HTTP method to be exploited (`GET` or `POST`). The resource URI contains the API method to be exploited, along with several configuration parameters, the API key (which identifies the entity performing the request) and the format of the output document.

FOAF Management. The *FOAF Management macro-area* consists of three modules (see Figure 1), namely *FOAF Controller*, *FOAF Request Handler* and *FOAF Document Handler*. The role of this macro-area is to extract information about the contacts of a user by exploiting FOAF [3] data sources. These allow the representation of a whole social network without the need of a centralized database. In fact, by exploiting this technology, it is possible to represent the information concerning a user account, along with the corresponding contacts and activities, through an RDF graph serialized as an XML document according to the W3C RDF/XML syntax. The working flow of this macro-area is as follows. First, *SN Parser* activates *FOAF Controller*. This is implemented as a thread and is in charge of coordinating the macro-area tasks. For this reason, it receives the SN-dependent query from *SN Parser* and generates the correct URL of the corresponding FOAF data source. Then, it activates *FOAF Request Handler*. This module performs an HTTP request to the specific social network through the module *SN FOAF Front-end* in such a way as to obtain the RDF/XML file. This file is then returned to *FOAF Controller*. After this, *FOAF Controller* activates *FOAF Document Handler* which exploits the JAVA JAXB Library to parse the RDF/XML file and to extract the needed information. This is returned by *FOAF Document Handler* to *FOAF Controller* which, in its turn, returns it to *SN Parser*.

XFN Management. This macro-area consists of three modules (see Figure 1), namely *XFN Controller*, *HTML Request Handler*, and *HTML Document Handler*. It is in charge of gathering information about existing `me` edges by exploiting the XFN technology [4]. Basically, XFN allows for the representation of the kind of relationship existing between two user accounts. This is obtained by empowering the set of values that the `rel` attribute of the HTML tag `<a>` (which represents a link) can assume. In our case, we focus on the value "me" (`rel='me'`) which indicates that the corresponding link represents a `me` edge. The working flow of this macro-area is as follows. First, *SN Parser* activates *XFN Controller*. It is implemented as a thread. This way, the tasks of the *XFN*, *FOAF* and *API Management macro-areas* can be executed in pipeline in such a way as to reduce the system running time. *XFN Controller* is the core of the *XFN Management* macro-area. First, it receives the SN-dependent query generated by *SN Node Solver*. Then, starting from this query, it creates the

correct URL identifying the Web resource storing the desired XFN information, and activates *HTML Request Handler*. This module performs an HTTP request to *SN HTML Front-end* to obtain the XFN-compliant HTML page. Afterwards, this page is returned to *XFN Controller*. After this, *XFN Controller* activates *HTML Document Handler* which parses the page specified in input and extracts the existing `me` edges from it. *HTML Document Handler* returns these edges to *XFN Controller* which, in its turn, returns them to *SN Parser*.

The SN Front-end macroarea. This macro-area consists of three modules, namely *SN API Front-end*, *SN FOAF Front-end*, and *SN HTML Front-end* (Figure 1). Each of these modules represents an entry point to the specific *SN Database*. This macro-area is reported in our system architecture only for the sake of completeness. As a matter of fact, its modules collect functionalities provided by the corresponding social networks.

3 Prototype and Performance Analysis

We implemented SNAKE as a Java library which can be integrated in any system needing information about social network users (e.g., a crawler, a search engine, a user profiling system, etc.). In order to provide a demo of its capabilities, we integrated it into a Web application, implemented by means of the JavaServer Pages and the Java Servlet technologies. This Web application runs on a Tomcat application server installed on a 2 Quad-Core E5440 processor and 16 GB of RAM with the CentOS 6.0 Server operating system. It is reachable at the address `http://ictsud.unirc.it:8080/snake-demo.html`. It allows a visitor to manually perform a request of information about the links of a social network user u. In particular, it requires the URL of the account of u and returns two lists, namely the list of the friends of u in the same social network and the lists of the accounts of u in other social networks, if any. The elements in the two lists are clickable links, which allows the visitor to activate a new request of information centered on the corresponding account. With the support of the SNAKE prototype we carried out an experiment aiming at measuring the performance of SNAKE, in particular its running time against the degree of the nodes provided in input. We carried out the experiment as follows:

- We run a BFS-based crawler to derive four SIS samples, which we obtained starting from a seed of Flickr, Friendfeed, LiveJournal and Advogato, respectively.
- We defined five degree intervals $I_1 ... I_5$. I_1 (resp., I_2, I_3, I_4 and I_5) comprised nodes with a degree belonging to the interval (0..100] (resp., (100..250], (250..500], (500..750], (750..∞)). We selected 100 nodes for each interval in such a way that each interval contained nodes of all the social networks mentioned above.
- We run SNAKE for each node and we measured the corresponding running time. Obtained results, averaged among the nodes of each interval, are reported in Figure 2 where we show the average running time of SNAKE as a whole, as well as the one of its API and FOAF components.

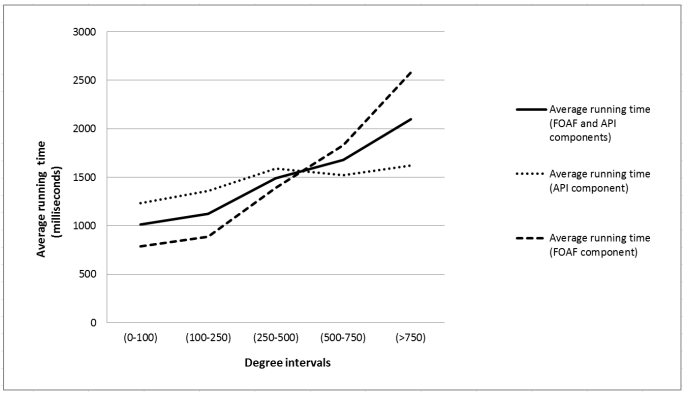


Fig. 2. Average running time of our system against node degree

From the analysis of this figure it emerges that the running time of our system has a linear trend. This is a very important result since it states that our system is efficient enough. In other words, if, in the future, the number of contacts per person tends to increase (as it appears plausible), our system is still capable of working with an acceptable increase of its running time.

As for another important result, we can observe that for low-degree nodes the FOAF component of SNAKE has a better performance than the API component. The opposite trend can be observed for high-degree nodes. This fact can be explained by considering that the FOAF document parsed by *FOAF Handler* to derive required information is unique, independently of the information amount represented in it (which is directly proportional to the contact number of the corresponding account and the degree of the corresponding node). When node degree is low, this document is small and, therefore, is managed quickly. As node degree increases, its dimension increases too, which makes its management more and more expensive. The way of proceeding for APIs is completely different; indeed, these last ones can operate parallelly.

4 Conclusion

In this paper, we have presented SNAKE, a system allowing the extraction of information from user accounts in a SIS and, hence, extremely useful for supporting community mining and people recommendations in this scenario. We have described the system architecture, as well as the modules composing it. Then, we have presented the prototype implementing it and an experiment devoted to analyze its performance. As for future work, first of all we plan to develop a further macro-area of SNAKE capable of deriving user account information by directly parsing the Web page associated with that account. This is necessary since some social networks do not provide any other way to query their data. After this, we plan to enrich our system in such a way as to make it

capable of constructing a profile for each user by merging the information stored in all her accounts in the social networks she is registered to.

References

1. Google Social Graph. `http://code.google.com/p/itswhoyouknow/wiki/SocialGraph`, 2012.
2. Online Identity Consolidator. `http://www.plaxo.com/info/opensocialgraph/`, 2012.
3. The Friend of a Friend (FOAF) project. `http://www.foaf-project.org/`, 2012.
4. XFN - XHTML Friends Network. `http://gmpg.org/xfn`, 2012.
5. A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi. Online team formation in social networks. In *Proc. of the International World Wide Web Conference (WWW 2012)*, pages 839–848, Lyon, France, 2012. ACM.
6. F. Buccafurri, G. Lax, A. Nocera, and D. Ursino. Crawling Social Internetworking Systems. In *Proc. of the International Conference on Advances in Social Analysis and Mining (ASONAM 2012)*, Istanbul, Turkey, Five-Pages Paper. Forthcoming. IEEE.
7. D.H Chau, S. Pandit, S. Wang, and C. Faloutsos. Parallel crawling for online social networks. In *Proc. of the International Conference on World Wide Web (WWW'07)*, pages 1283–1284, Banff, Alberta, Canada, 2007. ACM.
8. X. Cheng, C. Dale, and J. Liu. Statistics and Social Network of Youtube Videos. In *Proc. of the International Workshop on Quality of Service (IWQoS 2008)*, pages 229–238, Enschede, The Netherlands, 2008. IEEE.
9. P. De Meo, A. Nocera, G. Terracina, and D. Ursino. Recommendation of similar users, resources and social networks in a Social Internetworking Scenario. *Information Sciences*, 181(7):1285–1305, 2011. Elsevier.
10. J. Heer and D. Boyd. Vizster: Visualizing online social networks. In *Proc. of the IEEE Symposium on Information Visualization (INFOVIS 2005)*, pages 32–39, Minneapolis, MN, USA, 2005. IEEE.
11. OFCOM The independent regulator and competition authority for the UK communications industries. Social Networking: A quantitative and qualitative research report into attitudes, behaviours and use. `http://www.ofcom.org.uk/advice/media_literacy/medlitpub/medlitpubrss/socialnetworking/annex3.pdf`, 2009.
12. A. Mislove, M. Marcon, K.P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *Proc. of the ACM SIGCOMM International Conference on Internet Measurement (IMC'07)*, pages 29–42, San Diego, CA, USA, 2007. ACM.
13. Y. Okada, K. Masui, and Y. Kadobayashi. Proposal of Social Internetworking. In *Proc. of the International Human.Society@Internet Conference (HSI 2005)*, pages 114–124, Asakusa, Tokyo, Japan, 2005. Lecture Notes in Computer Science, Springer.
14. L.A.S. Pizzato and C. Silvestrini. Stochastic matching and collaborative filtering to recommend people to people. In *Proc. of the International Conference on Recommender Systems (RecSys'11)*, pages 341–344, Chicago, IL, USA, 2011. ACM.

Testing People To People Recommender in a Live Environment

Michael Meisel[1], Stefan Dahms, and Andreas Ittner

Hochschule Mittweida, 09648 Mittweida, Germany,
meisel@hs-mittweida.de

Abstract. Although recommender systems are very common in the online world today, there is relatively little work on testing these systems in a live environment. The quality of new recommendation algorithms is often measured by offline experiments on past data. In this work we run A/B tests with different people to people recommendation strategies and compare the results with the outcomes of an offline experiment.

Keywords: A/B Testing, People To People Recommendations, Evaluation

1 Introduction

Today the quality of recommendation strategies is often measured by offline experiments which have a lot of advantages. Detached from the live system one can get comparable and reproducible results without the limitations and requirements of a live system. Therefore a given business problem is transformed into a data mining problem. For example, if an online dating website wants to enhance the user experience with partner recommendations of high quality, they create experiments based on historical data of users who have found their counterparts. The results of these experiments are taken as a quality measure for the used algorithms. But as our work will show, the results of such experiments can't always be transferred to the live environment. The correct way to measure the quality of a recommender system regarding a business problem is to test it in the live environment. In section two we will explain the environment where the test had been done. In section three we will explain the basics of a proper A/B test with respect to questions like evaluation criterions, desired sample sizes and static significance. In section four we will present the different people to people recommendations we have tested and the results of the A/B tests.

Related work There is a lot of research in the field of recommender systems and controlled experiments. Analyzing an offline data set is the common practice in evaluating recommendation algorithms [3] [2]. There is also work where recommendation algorithms were tested in controlled experiments online [12]. On the other side Kohavi et. al [1] show that the results of online experiments are often unexpected. In our work we connect both types of recommender evaluation

with the goal to review the results from the offline experiments with a proper A/B test.

2 Environment

The environment is a matchmaking job portal with two different groups of users. Members of the first group (employers) offer jobs in the field of child care (live in nanny positions) and members of the second group (employees) can apply for the offered job positions. All group members (no matter what group they belong) are able to send standard contact requests to their counterparts. A contact request can be confirmed or rejected. A confirmed contact request is called contact. Only if a member has a contact and only if he/she is a premium member (i.e. paid membership) then this member is able to use the contact data of his/her counterpart for the purpose of individual communication. A user can become a standard member of the portal (free membership) by filling out a registration sheet. During the registration process a set of personal information is asked. This personal information (user features) can be used to support the matchmaking process between employers and employees.

3 Testing in Live Systems

An A/B test is a statistical test where two test groups are exposed to a different treatment. The aim is to determine whether there are statistically significant differences between the two groups regarding an evaluation criterion. The null hypothesis is, that there is no difference in the two groups. In practice, a test is considered statistically significant when it reaches a confidence of 95% and a power between 80-95% [4]. A confidence level of 95% means that 5% of the time we will reject the null hypothesis when it's true [5]. Power is the probability that we correctly reject a false null hypothesis [6]. To determine the required test cases at a desired confidence level of 95% and a power of 90%, we use the following formula [7]:

$$n = (4r\sigma/\Delta)^2 \tag{1}$$

where n is the number of test cases, r the number of test groups, σ is the standard deviation of the evaluation criterion, and Δ is the measured difference of the evaluation criterion.

3.1 Overall Evaluation Criterion (OEC)

The Overall Evaluation Criterion (OEC) is a quantitative metric that describes the objective of the experiment [8]. It is important to set the OEC before performing the test. Otherwise there is a risk of finding things in retrospect to be significant which came out by chance [9]. As mentioned, the aim of our work is to determine the quality of people to people recommendations on a business case and to compare them with the results of an offline evaluation. Our business

case is to increase the number of contact requests between the members through appropriate recommendations. Therefore, a recommendation will be considered good when a contact request is send after a member found the recipient trough this recommendation. When selecting the OEC some factors have to be taken into account. It is advisable to choose an OEC with a low variability, i.e. a low standard deviation, to minimize the required test sample size [10]. Probabilities (0%-100%) typically have a lower standard deviation than absolute numbers [10]. Therefore we decided to set the OEC as the probability that a user sends a contact request, after seeing a user profile via a recommendation (conversion). In addition we also looked at the proportion of profile views via recommendations to the entire profile views (proportion views).

3.2 User Splitting

As described in section two there are two different groups of members in the portal (employers/employees). When splitting the users into test group A and B it is important to ensure that the distribution of the two member groups is the same in both test groups as in the whole population, because it might be possible that an employer responds differently to the recommendations than an employee. We decided to make the two test groups equal in size because this maximizes the power of the test and minimizes the required test sample size [10]. At the beginning of the test the users are assigned to a test group. This assignment doesn't change for the whole test period. This ensures that the users are exposed to the same treatment over the entire period of the test. To assign a user to a test group we take into account to which member group he/she belongs. Afterwards we assign this user to the test group which has the least members of his/her member group. If both test groups have the same number of members of his/her member group he/she is randomly assigned to a test group. This method ensures that both test groups have the same size and distribution of member groups.

3.3 A/A Test

Prior the A/B tests we run an A/A test or null test [11]. In an A/A test both test groups are exposed to exactly the same treatment. The aim is to verify that there are no statistically significant differences between the groups. The A/A test is used to check the test setup and to find possible systematic errors, for example user splitting, group assignment, time delays. If there is a statistically significant difference between the two A groups then there is probably an error in the test setup.

As seen in Table 1 there are small differences between the two groups in the A/A test. But with the given sample size these differences are statistically not relevant and within the range of chance. We are aware that with the present sample size only differences from more than 20% can be found. Because of the

Table 1. Conversion of the A/A-Test

	group 0	group 1	sum
not converted view	718	467	1185
converted view	211	176	387
sum views via rcn	929	643	1572
conversion rate	22,71%	27,37%	24,62%
rel. deviation from mean	-7,74%	11,18%	

limited time available we decided that this is sufficient to avoid large systematic errors.

4 Experiments

First of all we will explain some terminology. The recommendations used in the portal are so called people to people recommendations. Unlike in traditional recommender systems from the field of e-commerce, which suggest items to users, in this portal a member can be both, user and item. We transformed the terminology of these classic recommender systems to our application. A user is the one who triggers an action and the item is the target of the action. Member i sends member j a contact request means that member i is the user and member j is the item. If member j confirms this request then j is the user and i is the item. So when we talk about a top seller, we mean that this member was most often the target of an action so he/she received the most contact requests.

4.1 Offline experiment

We designed an offline experiment to evaluate different recommendation algorithms. Therefore we used the data of 50k contact requests which have been sent in the past. We extracted one contact request from about 1000 different users and used this data as test set. The remaining 49k contact requests were used as training set. The aim was to predict the members who received a contact request from the users in the test set. After the training period we calculated for every user in the test set 6 recommendations and counted how often the recommendations contained the member who has received the request (correctly predicted request) The evaluation metric was the recall of sent contact requests which is the proportion of correctly predicted contact requests to the sum of correctly predicted contact requests and wrongly predicted contact requests.

$$recall = \frac{correctly\ pred.\ requests}{correctly\ pred.\ requests + wrongly\ pred.\ requests} \tag{2}$$

We compared 4 different approaches which are explained in detail later. The two collaborative filtering algorithms (CF) outperformed the random and the content-based approach in the offline evaluations (Table 2).

Table 2. Results Offline Evaluation

recommender	recall
Random	0,36%
Content-based	6,02%
CF (item-based)	15,82%
CF (user-based)	17,53%

4.2 Random Topseller vs. Collaborative Filtering (user-based)

In the first A/B test we decided to compare two methods which performed very differently in the offline evaluation, because big differences can be found faster (smaller test size is needed). So we chose the Random Topseller (RTS) and the user-based collaborative filtering approach (UCF). The calculation of the recommendations was done for both methods once a day. User-based means that the recommendations depend on the member which is logged in. For each member 6 recommendations have been calculated every day. These recommendations did not change during the whole day and the members saw the exact same 6 recommendations on every user profile they visited.

Random Topseller (RTS) The RTS method randomly chooses for each user 6 members from the list of the 100 members which received the most contact requests during the last 3 months.

User-based Collaborative Filtering (UCF) The UCF approach is about finding the n most similar users (peer group) and to recommend the members which were contacted by the peer group but not yet by the user. Two users are considered similar when they contacted the same members in the past. To compute the UCF we used a user-item matrix $A^{n \times n}$ with $A \in [0; 1]$ and n is the number of members in the portal. The contact requests of the past are stored in this matrix. The rows contain the users (senders) and the columns contain the items (recipients). If a member i sends a member j a contact requests then $a_{ij} = 1$ otherwise 0. To calculate the most similar users we used the cosine distance (Eq.3).

$$similarity = \frac{a_1 \cdot a_2}{\|a_1\|\|a_2\|} \tag{3}$$

Then we counted how often each member was contacted by the 50 most similar users. The value 50 for the size of the peer group had been found useful in the offline evaluation. The 6 most frequently contacted members by the peer group which the user had not contacted yet are the 6 recommendations for the user.

There is a clear difference between the two methods regarding the conversion rate (Table 3). In the RTS group the conversion rate was 17.05% compared to

Table 3. Experiment 1 - Results Conversion rate

	RTS	UCF	sum
not converted view	1080	571	1651
converted view	222	344	566
sum views via rcn	1302	915	2217
conversion rate	17,05%	37,60%	25,53%
rel. deviation from mean	-33,21%	47,26%	

Table 4. Experiment 1 - Results Proportion Views

	RTS	UCF	sum
normal view	29114	27767	56881
view via rcn	1302	915	2217
sum views	30416	28682	59098
proportion views via rcn	4,28%	3,19%	3,75%
rel. deviation from mean	14,11%	-14,96%	

the UCF group with 37.60%. According to our test criteria 836 samples are required to conduct that this difference is statistically relevant. The assumption of the offline test can be confirmed. We can say with 95% confidence that the recommendations of the UCF approach lead to a significantly higher number of contact requests than the RTS approach. Considering the proportion of views via recommendations (Table 4) it seems that the RTS approach performs a little bit better than UCF. This indicates that the RTS recommendations have been clicked more often than the recommendations in the other group. But the difference is very small (4,28% to 3,19%) and with respect to our test criteria it could easily be random.

4.3 Content-based vs. Collaborative Filtering (item-based)

Compared to the A/A test the proportion of profile views via recommendations in experiment one have decreased. We assume that the reason is the static nature of the user-based recommendations as a user receives at best only 6 new recommendations per day. So for the next experiment we selected two item-based approaches. In this context item-based means that the displayed recommendations are not related to the member which is logged in but they are related to the profile that will be looked at. For each profile, the 6 most similar profiles are calculated. The result is that users can receive 6 new recommendations on each profile they look at and so they are able to see more recommendations in total. For the second test we chose a content-based approach and a collaborative filtering method again. In other work there are studies about the different

results and benefits of both approaches in the field of people to people recommendations [12]. In our offline evaluation the collaborative filtering method was superior (Table 2).

Content-based (CB) The content-based approach compared corresponding fields, which are filled by the members during the registration process on the portal. We used the same cosine similarity as used for collaborative filtering to find similar profiles. (Eq.3). An unweighted and a weighted strategy have been applied. The weighted strategy gave more importance to fields that were believed to be more significant for a good match. The fields actually used were selected by consulting the portal manager. We selected only a small subset of the fields available e.g. country, age or gender. Out of the experience gained so far from user feedback these fields seemed to be the most important for an exact match. Comparing numerical attributes like age was straight forward. For comparison of categorical attributes like country each occurring characteristic was transformed into a field name after the characteristic and got a binary value of either one or zero.

Item-based Collaborative Filtering (ICF) To determine similar profiles based on interaction data we assume that the profiles of two members are similar if:

- both members have received a request from the same 3rd member.
- both members sent the same 3rd member a contact request with positive response, i.e. a confirmed contact request.
- one of the two members has received a request from the same 3rd member the other already sent a contact request which was confirmed.

The calculation has been performed on the user-item matrix $A^{n \times n}$ with $A \in [0; 1]$ and n is the number of members in the portal. Element $a_{ij} = 1$ means a member i sent a member j a contact request or j confirmed a request from i. The similarity between the column vectors is computed again by the cosine distance (Eq.3). The 6 most similar members are the 6 recommendations for the visited profile.

Table 5. Experiment 2 - Results Conversion rate

	CB	ICF	sum
not converted view	2664	2691	5355
converted view	868	950	1818
sum views via rcn	3532	3641	7173
conversion rate	24,58%	26,09%	25,35%
rel. deviation from mean	-3,04%	2,95%	

Table 6. Experiment 2 - Results Proportion Views

	CB	ICF	sum
normal view	19861	21818	41679
view via rcn	3532	3641	7173
sum views	23393	25459	48852
proportion views via rcn	15,10%	14,30%	14,68%
rel. deviation from mean	2,83%	-2,60%	

Surprisingly in the A/B test both the conversion rate as well as the proportion of views via recommendations show only very small differences (Table 5/6). The differences are too small to be considered statistically significant within the given test size. The assumption from the offline evaluation that the collaborative filtering approach performs much better than the content-based method (Table 2) could not be confirmed. The proportion of views via recommendations are much higher in this experiment compared to the first test. As mentioned before we assume that this is due to the fact that the users in this experiment saw much more recommendations. Based on the results of the second experiment we conclude that the offline evaluation we made prior the A/B tests could not provide reliable information about the quality of the tested recommendation strategies in the live environment regarding the business case.

5 Summary

The aim of our work was to review the results of an offline evaluation of people to people recommendations in an online A/B test. We have shown how to design a proper A/B test and which aspects are important. We concluded that it was not possible to make reliable assumptions about the performance of the different recommendation strategies in the live environment from the results in the offline evaluation. Especially the big differences in the proportion of views via recommendations between the two A/B tests and the not existing difference between the content-based approach and the collaborative filtering method in experiment two were very surprising. We strongly recommend to run A/B tests for evaluating recommender systems when its possible. Modeling a business case in a proper data mining problem is a challenging task. Therefore we see the need for future work on how offline experiments should be designed to draw reliable conclusions for the real world application of recommender systems.

References

1. Ron Kohavi, Alex Deng, Brian Frasca, Roger Longbotham, Toby Walker, Ya Xu: Trustworthy online controlled experiments: five puzzling outcomes explained. KDD 2012: 786-794

2. R. Burke. Hybrid recommender systems: Survey and experiments. User Modeling and User-Adapted Interaction, 12(4): 331-370, 2002.
3. Groh, G., & Ehmig, C. 2007. Recommendations in Taste Related Domains: Collaborative Filtering vs. Social Filtering. Proc. ACM Group'07. 127-136.
4. Box, George E. P., Hunter, J. Stuart and Hunter, William G. Statistics for Experimenters: Design, Innovation, and Discovery. 2nd. s.l. :John Wiley & Sons, Inc, 2005.
5. Park, Hun Myoung. 2008. Hypothesis Testing and Statistical Power of a Test. Working Paper. The University Information Technology Services (UITS) Center for Statistical and Mathematical Computing, Indiana University. http://www.indiana.edu/ statmath/stat/all/power/index.html
6. Greene, William H. 2000. Econometric Analysis, 4th ed. Upper Saddle River, NJ: Prentice Hall.
7. Wheeler, Robert E. 1974. Portable Power, Technometrics, Vol. 16. http://www.bobwheeler.com/stat/Papers/PortablePower.PDF.
8. Roy, Ranjit K. Design of Experiments using the Taguchi Approach: 16 Steps to Product and Process Improvement. s.l. :John Wiley & Sons, Inc, 2001.
9. Keppel, Geoffrey, Saufley, William H. and Tokunaga, Howard. Introduction to Design and Analysis. 2nd. s.l. :W.H. Freeman and Company, 1992.
10. Ron Kohavi , Randal M. Henne , Dan Sommerfield, Practical guide to controlled experiments on the web: listen to your customers not to the hippo, Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, August 12-15, 2007, San Jose, California, USA
11. Peterson, Eric T. Web Analytics Demystified: A Marketer's Guide to Understanding How Your Web Site Affects Your Business. s.l. :Celilo Group Media and Cafe Press, 2004.
12. J. Chen, W. Geyer, C. Dugan, M. Muller, and I. Guy. Make new friends, but keep the old: recommending people on social networking sites. In CHI '09, pages 201-210, New York, 2009. ACM.

Identifying Topical Twitter Communities via User List Aggregation

Derek Greene, Derek O'Callaghan, Pádraig Cunningham

School of Computer Science & Informatics, University College Dublin
{derek.greene,derek.ocallaghan,padraig.cunningham}@ucd.ie

Abstract. A particular challenge in the area of social media analysis is how to find communities within a larger network of social interactions. Here a community may be a group of microblogging users who post content on a coherent topic, or who are associated with a specific event or news story. Twitter provides the ability to curate users into lists, corresponding to meaningful topics or themes. Here we describe an approach for crowdsourcing the list building efforts of many different Twitter users, in order to identify topical communities. This approach involves the use of ensemble community finding to produce stable groupings of user lists, and by extension, individual Twitter users. We examine this approach in the context of a case study surrounding the detection of communities on Twitter relating to the London 2012 Olympics.

1 Introduction

A wide variety of community finding techniques have been proposed in the literature, with recent research focusing on the challenge of identifying overlapping communities [1]. In the case of microblogging data, researchers have been interested in the identification of communities of users on Twitter, who produce tweets on a common topic, who belong to the same demographic, or who share a common ideological viewpoint [2]. These approaches have generally relied on explicit views of the Twitter network, such as follower relations or retweets.

Twitter users can organise the accounts that they follow into Twitter *user lists*. These lists are used in a variety of ways. In some cases they may correspond to personal lists of a given user's friends and families, but frequently lists are employed to group together Twitter accounts based on a common topic or theme. In this way, every Twitter user can effectively become a community curator. Notably, journalists from news organisations such as The Telegraph and Storyful curate lists relevant to a given news story or event, as a means of monitoring breaking news. Recently, Kim *et al.* and García-Silva *et al.* [3, 4] both discussed the potential of user lists to provide latent annotations for Twitter user profiles.

Our primary goal here is to demonstrate that topical communities can be identified by harnessing the "crowd-sourced" list building efforts of a large base of Twitter users. In Section 3, we show that this can be done by constructing a graph based on the similarity of user list memberships, and then using an *ensemble community finding* approach to find robust, overlapping groups of lists

within this graph, from which user communities can be derived. We use stability information derived from the ensemble as a proxy for the reliability of the communities. In Section 4, we evaluate the proposed techniques on a case study relating to coverage of the London 2012 Olympics on Twitter.

2 Related Work

Many researchers have become interested in exploring the network structure within the Twitter network, given the potential for Twitter to facilitate the rapid spread of information. Java *et al.* [2] provided an initial analysis of the early growth of the network, and also performed a small-scale evaluation that indicated the presence of distinct Twitter user communities. Kwak *et al.* [5] performed an evaluation based on a sample of 41.7m users, studying aspects of the network such as: identifying influential users, information diffusion, and trending topics. Typically researchers have focused either on Twitter users from the perspective of the content that they produce, or in terms of explicit network representations based on follower relations or retweeting activity [5]. However, preliminary work by Kim *et al.* [3] suggested that latent groups and relations in Twitter data could be extracted by examining user list data. Wu *et al.* [6] suggested that user list memberships could be used to organise users into a predefined set of categories: celebrities, media, organisations, and blogs. García-Silva *et al.* [4] described approaches for extracting semantic relations from user lists, by constructing relations between co-occurring keywords taken from list names.

Many algorithms have been proposed to identify communities in graphs, based on different combinations of objective functions and search strategies [1]. A widely-employed algorithm in this area is OSLOM (Order Statistics Local Optimization Method), introduced by Lancichinetti *et al.* [7]. Kwak *et al.* [8] observed that many community detection algorithms can produce inconsistent results, due to stochastic elements in their optimisation process. Lancichinetti & Fortunato [9] demonstrated that this also applied to OSLOM, and proposed an ensemble approach to generate stable results from a set of partitions. In the more general cluster analysis literature, *ensemble clustering* methods have been developed to address similar issues. These typically involve generating a diverse set of "base clusterings", which are aggregated to produce a consensus solution [10, 11].

3 Methods

In this section, we introduce an approach that aggregates user list information to generate communities. Firstly, we describe the construction of a graph representation of user lists, based on their membership overlaps. Then in Section 3.2 we describe an ensemble approach to identify overlapping groups of user lists. The stability of these groups is assessed as described in Section 3.3, and the selection of community labels is discussed in Section 3.4. Finally, the derivation of corresponding communities for individual users is discussed in Section 3.5.

3.1 User List Graph Construction

We construct a graph G of l nodes, where each node represents a distinct Twitter user list L_x. A weighted edge exists between a pair of lists if they share users in common. Rather than using the raw intersection size between a pair, we make allowance for the significance of the intersection size relative to the size of the two lists, and the total number of users assigned to lists n. For a pair (L_x, L_y), we compute a p-value to indicate the significance of the probability of observing at least $|L_x \cap L_y|$ users from L_x within another list of size $|L_y|$:

$$PV(L_x, L_y) = 1 - \sum_{j=0}^{|L_x \cap L_y|-1} \frac{\binom{|L_x|}{j}\binom{n-|L_x|}{|L_y|-j}}{\binom{n}{|L_y|}} \tag{1}$$

To improve interpretability, we compute the associated log p-value:

$$LPV(L_x, L_y) = -\log\left(PV(L_x, L_y)\right) \tag{2}$$

where a larger value is more significant. We consider Eqn. 2 as a measure of the similarity between a pair of user lists, corrected for chance. To further increase the sparseness of the graph, we remove edges with weights $LPV < \rho$ for a weight threshold ρ. Increasing the value of ρ will result in an increasingly sparse graph.

3.2 Combining Overlapping Communities

We will naturally expect that different topical communities will potentially overlap with one another. To identify communities of lists, we apply the OSLOM algorithm which has been shown to out-perform other community finding approaches [7]. However, as noted in [9], OSLOM can produce unstable results.

Following the CSPA ensemble aggregation approach [10], and the method for combining network partitions [9], we now describe an approach for generating and combining an ensemble of overlapping community sets. Given an initial user list graph G, we construct a symmetric $l \times l$ *consensus matrix* $\mathbf{M}$. For the purpose of generating a collection of r *base community sets*, we apply the OSLOM algorithm [7] using a different initial random seed for each run. Motivated by the notion an ensemble of weak clusterings [11], we use the "fast" configuration of OSLOM, which uses a minimal number of optimisation iterations.

After generating a base community set, for each unique pair of nodes (L_x, L_y) in network, we compute the Jaccard similarity between the sets of community labels assigned to those nodes by OSLOM. If the pair are not both co-assigned to any community, the score is 0. If the pair are present in all communities together, the score is 1. However, unlike the binary approach of [9], if the pair are present in some but not all communities together, the Jaccard score will reflect this. See Fig. 1 for examples. In the case of non-overlapping partitions, the score will reduce to the binary scoring used in [9]. After computing all Jaccard scores, we increment the corresponding matrix entries in $\mathbf{M}$.

Once all r base community sets have been generated, $\mathbf{M}$ is normalised by $1/r$ to give a matrix with entries $\in [0, 1]$. To find the *consensus communities*,

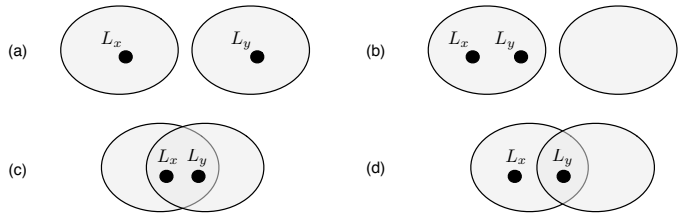


Fig. 1. Four different cases of computing the Jaccard similarity between the sets of community labels assigned to two nodes L_x and L_y. The Jaccard scores respectively are: (a) 0.0, (b) 1.0, (c) 1.0, (d) 0.5.

we follow a similar approach to that used by [9]. We construct a new undirected weighted graph such that, for every unique pair of nodes (L_x, L_y), we create an edge with weight M_{xy} if $M_{xy} \geq \tau$. The threshold parameter $\tau \in [0, 1]$ controls the sparsity of the graph. We then apply OSLOM to this graph to produce a final grouping of user lists.

3.3 Evaluating Community Stability

When applying community detection, often we may wish to examine the most reliable or robust communities with the strongest signals in the network. Here, we rank the consensus communities generated as described in Section 3.2, based on the cohesion of their members with respect to the consensus matrix $\mathbf{M}$. A more stable consensus community will consist of user lists which were frequently co-assigned to one or more communities across all r base community sets.

For a given consensus community C of size c, we compute the mean of the values M_{xy} for all unique pairs (L_x, L_y) assigned to C; this value has the range $\in [0, 1]$. We then compute the mean expected value for a community of size c as follows: randomly select c unique nodes from G, and compute their mean pairwise score from the corresponding entries in $\mathbf{M}$. This process is repeated over a large number of randomised runs, yielding an approximation of the expected stability value. We then employ the widely-used adjustment technique introduced by [12] to correct stability for chance agreement:

$$\text{CorrectedStability}(C) = \frac{\text{Stability}(C) - \text{ExpectedStability}(C)}{1 - \text{ExpectedStability}(C)} \tag{3}$$

A value close to 1 indicates a highly-stable community, while a value closer to 0 is indicative of a weak community that appeared intermittently over the r runs. We rank all consensus communities based on their values for Eqn. 3.

3.4 Selecting Community Labels

To summarise the content of a consensus community, we aggregate the meta-information associated with all lists assigned to that community. Specifically, we

construct a bag-of-words model, where each user list is represented by unigrams and bigrams tokenised from the list's name and description. Single stop-words are removed, and terms are weighted using log-based TF-IDF. For each community, we then compute the centroid vector corresponding to the mean vector of all lists assigned to that community. To generate descriptive labels for the community, we subtract the mean vector of **all user list vectors** from the community centroid vector, and rank terms in descending order based on the resulting weights. The top ranked terms are used as community labels.

3.5 Deriving User Memberships

The consensus communities generated using the method proposed in Section 3.2 can potentially provide us with an insight into the overall topics in a Twitter corpus. However, it will often be useful to assign community memberships to individual users. We can readily produce this by using the list groupings in conjunction with the original user list membership information.

For a given consensus community C of size c, we examine the memberships of all lists in the community. We consider the assignment of a user u_i to each $L_x \in C$ as being a vote with weight $1/c$ for u_i belonging to the overall community. The total membership weight for u_i is therefore given by the fraction of lists $L_x \in C$ containing u_i. Membership weights for all communities are computed in this way. We can also rank the importance of users in a given community by sorting users by weight in descending order. To produce a final set of user communities, we only include a user in a community for which the user has a membership weight $\geq \mu$, based on a membership threshold $\mu \in [0, 1]$.

4 Evaluation

4.1 Data Collection

To evaluate the proposed community finding methods, we constructed a dataset based on a list of 499 users curated by The Telegraph, which covers athletes, journalists, and organisations involved in the London 2012 Summer Olympics[1]. Initially, for each user we retrieved up to their 200 most recent user list assignments. From this initial pool of lists, we then retrieved list memberships for 10,000 randomly selected lists of size ≥ 5 and containing at least 2 core list users. This yielded a dataset containing a total of 44,484 individual list membership records, where the average number of lists per user was 89. The most frequently-listed user was *@andy_murray*, assigned to 1,931 different lists.

4.2 Community Detection

Using the approach described in Section 3.1, we constructed a user list graph based on membership information for the 499 users. To limit the density of the

[1] `http://twitter.com/#!/Telegraph2012/london2012`

graph, we use a weight significance threshold of $\rho = 6$ (*i.e.* user list overlaps are considered as significant for $LP \leq 1^{-6}$). This resulted in a graph containing 4,948 nodes representing user lists, with 749,062 weighted edges between them.

To generate an ensemble of base community sets, we apply OSLOM as described in Section 3.2 for 100 random runs, selecting the lowest level of the hierarchy as the solution for each run. The average number of non-singleton communities in each run was 157. Combining the base community sets yielded a consensus matrix containing $\approx$ 11.8m non-zero values. We examined a range of threshold values $\tau \in [0.1, 0.5]$, and selected a threshold $\tau = 0.2$ to generate consensus communities in order to maximise coverage over user lists, while also reducing the density of the consensus graph. Applying OSLOM to the sparse graph of $\approx$ 1.5m values produced a total of 94 consensus communities, considerably lower than the average base community count. Finally, user communities were derived using a low membership threshold $\mu = 0.1$ to maximise the number of core users assigned to communities. In total, 416 core users were assigned to at least one community, with 362 users assigned to multiple communities.

Table 1 shows the top 15 communities, arranged in descending order by their stability score, as defined in Eqn. 3. The table shows the size of each community (in terms of both number of lists and users assigned), the top text labels selected for each community, and the three highest-weighted users. We observe that the most stable communities generally correspond to communities of users involved in specific, "niche" sports (*e.g.* badminton, BMX racing, fencing). In these cases, the top-weighted users correspond to either British Olympic athletes competing in these sports, or accounts of the official British organisations

Table 1. Top 15 user list communities, arranged in descending order by stability score.

Score	Lists	Users	Top Labels	Top Users
1.00	17	14	badminton, badminton players, badders	@Jennywallwork, @Nath_Robertson, @ChrisAdcock1
1.00	5	5	bmx, bmx racing, bmx atlēti	@ShanazeReade, @liamPHILLIPS65, @bloomy181
1.00	32	11	sailing, sailors, olympic	@SkandiaTeamGBR, @AinslieBen, @matchracegirls
1.00	19	24	fencing, fencers, individuele schermers	@britishfencing, @CBennettGBR, @LaurenceHalsted
1.00	6	5	triathlon, machines, swim run	@AliBrownleetri, @jodieswallow, @MarkCavendish
1.00	5	21	scots, red sky, 2014	@mj88live, @RobbieRenwick, @Euan_Burton
1.00	22	4	wielrennen, ciclismo, cycling	@GeraintThomas86, @UCI_cycling, @MarkCavendish
0.98	5	7	track, field, track field	@allysonfelix, @TysonLGay, @tiffofili
0.98	48	19	rowing, rowers, gb rowing	@andrewthodge, @ZacPurchase, @MarkHunterGB
0.97	14	22	diving, tuffi, olympic diving	@PeterWaterfield, @matthew_mitcham, @toniacouch
0.96	36	44	hockey, hockey players, field hockey	@AlexDanson15, @RichM6, @jfair25
0.96	12	21	canoe, canoeing, canoe slalom	@GBcanoeing, @PlanetCanoe, @edmckeever
0.93	5	5	actors athletes, internet stars, athletes tmz	@usainbolt, @ShawnJohnson, @MichaelPhelps
0.92	27	13	judo, judo clubs, judo related	@BritishJudo, @USAJudo, @IntJudoFed
0.87	6	7	runners, hardlopen, runners world	@Mo_Farah, @paulajradcliffe, @KenenisaBekele

Table 2. Validation scores achieved relative to 18 "ground truth" categories.

Category Name	Category Size	Precision	Recall	F1
judo	20	1.00	0.65	0.79
basketball	26	1.00	0.50	0.67
rowing	44	1.00	0.43	0.60
athletics	50	1.00	0.22	0.36
cycling	28	1.00	0.14	0.25
hockey	47	0.98	0.91	0.95
diving	23	0.95	0.91	0.93
equestrianism	18	0.94	0.83	0.88
fencing	23	0.88	0.91	0.89
sailing	16	0.82	0.56	0.67
gymnastics	24	0.77	0.42	0.54
canoeing	22	0.76	0.73	0.74
beach-volleyball	12	0.55	1.00	0.71
boxing	22	0.55	0.55	0.55
swimming-syncrho	16	0.33	0.13	0.18
weightlifting	6	0.20	0.17	0.18
archery	17	0.20	0.06	0.09
waterpolo	22	0.05	0.05	0.05

for these sports. Interestingly, we also see some unexpected communities with high stability - a community around the Glasgow 2014 Commonwealth Games, and a community of celebrities which includes "elite" user accounts with hundreds of thousands of followers (*e.g. @usainbolt*, *@MichaelPhelps*). As stability decreases, we observed that communities become less homogeneous, covering highly-popular sports (*e.g.* football, basketball), or containing users and lists related to several sports. This suggests that the proposed stability provides a useful measure of the homogeneity of topical content for Twitter communities. Many of the top labels selected for communities are multi-lingual. For instance, the label for the "cycling" community in Table 1 contains terms in Dutch, Italian, and English. Unlike in certain textual analyses of tweets, the use of list membership information allows us to identify groups of users in a language-agnostic manner.

4.3 External Validation

To validate the consensus user communities that were identified by aggregating list information, we use a set of fine-grained Olympics lists also produced by The Telegraph[2], consisting of Twitter users associated with individual sports (*e.g.* "archery", "equestrianism"). This provided us with 18 external "ground truth" categories, covering 423 of the 499 users in the dataset.

We computed *precision*, *recall*, and *F1* scores for all communities, and subsequently matched categories to communities based on precision. Table 2 shows the resulting scores for all categories, arranged in descending order by precision. Communities produced by user list aggregation allowed us to identify eight categories with precision ≥ 0.9, while generally maintaining high recall. Only in the case of four categories did the proposed approach lead to precision and recall scores of both ≤ 0.5. Subsequent examination of the data suggests that list information was relatively sparse for these categories, and that the users were generally assigned to more generic lists (*e.g.* "aquatics" for "waterpolo").

[2] `http://twitter.com/#!/Telegraph2012/lists`

5 Conclusions

In this paper, we have presented initial work on the idea of identifying topical communities on Twitter by aggregating the "wisdom of the crowds", as encoded in the form of user lists. We show that this information can be mined to identify and label coherent overlapping clusters of both lists and users. We intend to expand this study by examining the identification of other types of user groups within Twitter (*e.g.* academic communities), and by comparing these groups with those mined from other views of the data (*e.g.* follower networks, tweet content).

While the evaluation in this paper used a fixed network of users, a similar approach could be applied to identify topical sub-communities around trending terms or hashtags. Also, in some cases a user will not have been assigned to any user lists. We suggest that a classification process, using an alternative network view (*e.g.* follower links) could be used to assign such users to communities.

Acknowledgements. This research was supported by Science Foundation Ireland Grant 08/SRC/I1407 (Clique: Graph and Network Analysis Cluster).

References

1. Fortunato, S.: Community detection in graphs. Phy. Rep. **486**(3-5) (2010) 75–174
2. Java, A., Song, X., Finin, T., Tseng, B.: Why we Twitter: understanding microblogging usage and communities. In: Proc. 9th WebKDD and 1st SNA-KDD Workshop on Web Mining and Social Network Analysis. (2007) 56–65
3. Kim, D., Jo, Y., Moon, I.C., Oh, A.: Analysis of Twitter lists as a potential source for discovering latent characteristics of users. In: Workshop on Microblogging at the ACM Conference on Human Factors in Computer Systems (CHI'10). (2010)
4. García-Silva, A., Kang, J., Lerman, K., Corcho, O.: Characterising emergent semantics in Twitter lists. The Semantic Web: Research and Applications (2012) 530–544
5. Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a social network or a news media? In: Proc. 19th Int. Conf. on World Wide Web. (2010) 591–600
6. Wu, S., Hofman, J., Mason, W., Watts, D.: Who says what to whom on Twitter. In: Proc. 20th International Conference on World Wide Web, ACM (2011) 705–714
7. Lancichinetti, A., Radicchi, F., Ramasco, J., Fortunato, S., Ben-Jacob, E.: Finding statistically significant communities in networks. PLoS ONE **6**(4) (2011) e18961
8. Kwak, H., Eom, Y.H., Choi, Y., Jeong, H., Moon, S.: Consistent Community Identification in Complex Networks. ArXiv e-prints (October 2009)
9. Lancichinetti, A., Fortunato, S.: Consensus clustering in complex networks. Sci. Rep. **2** (03 2012)
10. Strehl, A., Ghosh, J.: Cluster ensembles - a knowledge reuse framework for combining partitionings. In: Proc. Conference on Artificial Intelligence (AAAI'02), AAAI/MIT Press (2002) 93–98
11. Topchy, A., Jain, A., Punch, W.: Combining multiple weak clusterings. In: Proc. 3rd IEEE International Conference on Data Mining (ICDM'03). (2003) 331–338
12. Hubert, L., Arabie, P.: Comparing partitions. J. Classification (1985) 193–218

Social Citation: Finding Roles in Social Networks. An Analysis of TV-Series Web Forums

Nikolay Anokhin[1], James Lanagan[1], and Julien Velcin[2]

[1] Technicolor
1, avenue de Belle Fontaine - CS 17616
35576 Cesson-Sévigné Cedex, France
{nikolay.anokhin,james.lanagan}@technicolor.com
[2] ERIC Lab
5 av. Pierre Mendès-France
69676 Bron Cedex, France
{julien.velcin}@univ-lyon2.fr

Abstract. In this paper we present preliminary work studying the interactions of a community of focussed forum users and their discussions around several television series. We use k-means clustering and a number of novel citation-analysis inspired measures to perform bottom-up role detection on this community of TV fans, and show that these emergent roles correspond well with the positions assigned to users using traditional graph-based measures.

Keywords: Citation Analysis, Role Detection, Social Network Analysis

1 Introduction

When looking to spread information about new products or services, companies seek to find the most influential or connected people within their target community so as to achieve the greatest return for any investment. Citation analysis is an interesting approach to the problem of role identification as it allows us to build on the inherent ideas of importance of the work of an author. The primary goal of this work is to bring together existing approaches from citation analysis and machine learning so as to discover the different roles which forum users play. The major contributions of this paper are two-fold:

- We propose two measures adapted from the citation analysis domain – Node g-Index and Catalytic Power – and combine these with 2 existing measures – Cross-Topic Entropy and Generalised Degree – so as to quantify the interaction and importance of users within a community of forum users.
- By considering the contributions of users to a forum within a sliding time window, we are able to perform unsupervised clustering on the temporal representations of users to assign a principal (most representative) role. We

show that the assigned roles correspond well with the positions assigned to users using traditional graph-based measures.

In the next section we shall detail some of the related work before describing the collection and statistics of our experimental corpus. Section 4 details the new adapted measures that we have created so as to characterise our forum users. The experiments performed in this work are detailed in Section 5. Finally we discuss our conclusions and present some future directions for further research.

2 Related Work

We see the study of importance as a side-effect of social role detection. As with much past work [1, 2], we shall focus on the identification of 'important' actors from within a network. This does however provide a good starting point for future work on role categorisation and identification. There has been relative little work on studying roles through time or the dynamics of social roles [3]. We look to incorporate the temporal aspect of roles into our work by examining the interactions of users as a function of the current state of the network. We will not focus on the dynamics of role attribution but feel that this provides an important distinction from current approaches.

Garfield [4] noted many reasons for the citation of articles within a work that are strongly linked to the reasoning behind online conversations. We wish to use the approaches from citation analysis to provide a measure of interaction and importance to our forum users. This differs from purely graph-based measures as we intend to take into account only those interactions that have been judged sufficiently interesting.

3 Corpus Creation

Our corpus was created by crawling the TWOP[3] website. The website is designed for entertainment content providing forums for communities to converge and discuss TV shows, and contains dedicated sub-fora each focussed on a single television series. Each sub-forum contains many threads of conversation allowing us to tag each thread in our corpus with a single series of interest. We focus our analysis on a year of forum posts discussing 6 television series each having their own dedicated sub-forum. The shows were featured in the "Top Shows" categories (containing 8-10 television series) on the site (Table 1). A targeted crawl of the 6 sub-forums of interest retrieved 58,994 posts created by 7,066 authors. This number is different from that in Table 1 as some authors are active in more than one sub-forum; we use this to our advantage when identifying the activity profile of users.

[3] http://forums.televisionwithoutpity.com/

Table 1. Breakdown of corpus by sub-forum.

Forum	Genre	Threads	Posts	Authors
American Idol	Reality TV	40	15,907	2,354
Dexter	Suspense	44	3,296	596
Grey's Anatomy	Drama	52	12,319	1,926
House	Drama	58	11,920	1,371
Mad men	Drama	27	2,943	471
The Office	Comedy	71	12,609	1,692
Totals		292	58,994	8,410

3.1 Corpus Preparation

After crawling the forum, it was necessary to reconstruct the threads of discussion. We build a user network that takes into account communications between users: the more reply-response pairs there are between users A and B, the stronger their relationship. Having this idea in mind, we build our user network as an undirected weighted graph[4].

TWOP uses a 'quote' mechanism meaning that the quoted text of a message appears before the text of a reply. We used a series of regular expressions, as well as Levenshtein distance (95% overlap) to detect the parent-child relationship within the past 20 in-thread messages. This threshold was chosen empirically as it provided us with high retrieval. We manually checked the efficacy of the proposed thresholds across 10% of the corpus and found a retrieval rate 100% for all quoted parent texts.

4 Measuring Influence

Influence may be defined as *"the act or power of producing an effect without apparent exertion of force or direct exercise of command"*. Several *node centrality* measures developed in graph theory are traditionally used to identify the "most important" actors in a social network. They assume that important authors occupy central positions in the network graph: *degree centrality*, *betweenness centrality* and *closeness centrality*. As stated later in Section 5, we found that the highly-ranked nodes according to these measures were clustered together by our own measures.

Citation analysis is an early form of linkage analysis [4]. In this context, we shall use messages as analogous to publications. The most widely accepted citation analysis metrics are Hirsch's *h-index* [5], as well as the *g-index* which is a direct variant of the former. It considers only those items (the H-CORE) that are significant enough to have received a predefined number of replies/citations:

[4] Although originally based on a directed graph, we shall perform citation analysis on an undirected graph as the communications/citations often form part of a larger two-way conversation

> An author has an **h-index** of h if h of his total contributions have received at least h citations each.

The g-index was an improvement on this again as it also takes into account the distribution of the items within the h-core.

> An author has a **g-index** g if g is the highest rank such that the top g contributions have, together, at least g^2 replies. This also means that the top $g+1$ messages have less than $(g+1)^2$ replies.

4.1 Social Citation

In the following sections we describe 4 features (including 2 novel features adapted from the citation analysis domain) selected to best represent the different aspects of a forum user with regards to their overall output and interactions within the community. By using these features we hope to identify the roles that are played by users within the community (forum) [1]. The approach is exploratory: the idea is to let the roles' typology emerge using different kinds of measures.

Generalised Degree The GENERALISED DEGREE of a node A is defined as $D(A) = \sum_{B \in N(A)} W(A, B)$ where $N(A)$ is the set of neighbours of A. $W(A, B)$ is the number of communications between A and B. This feature expresses how active the user is taking into account all communications of the user with his neighbours. On a directed graph this would be equivalent to the combined in and out degrees.

Node g-Index This feature evaluates how active neighbours of the node are. G-INDEX is calculated as the highest number g of neighbours such that sum of their generalised degrees is g^2 or more.

Catalytic Power of Users The CATALYTIC POWER of a message k reflects the amount of reaction from other users caused by the message: messages with high catalytic power produce (catalyse) large discussions. In the context of forum discussions, we use the number of direct replies, c_k, to estimate this power. For a user we have a list of catalytic power of her messages $\hat{c} = c^1_{k_1}, c^2_{k_2}, \ldots, c^n_{k_n}$. The *catalytic power of a user* is consequently defined as the sum of powers from the h-core of $\hat{c}$: $C = \sum_{i=1}^{h} c^i_{k_i}$

Cross-Topic Entropy The idea of using ENTROPY to measure user's focus on a particular topic has been used previously [6]. Let us consider a user who posted n_i messages across all the threads that discuss topic i (in our case the name of the discussed show was considered a topic). Let $n = \sum_i n_i$, then focus of a user is defined as $F = \sum_i -\frac{n_i}{n} \log \frac{n_i}{n}$. This measure helps to distinguish between users who contribute to many topics across the forum from users who focus on a single topic (e.g. fans).

4.2 Temporal Aspects of Roles

Although analysis of users' features obtained from the whole data set (the entire calendar year 2007) reveals general patterns of interactions between users, it is also important to consider how values of those features evolve over time. We observe weekly peaks of activity on the broadcast dates of new episodes of each series, and almost no activity in summer when none of the six TV shows are broadcast. In addition, activity of each user is far from uniform (see Figure 1). In the current work we shall assign a user to their most representative role, but we do so by taking into account their weekly role within the community.

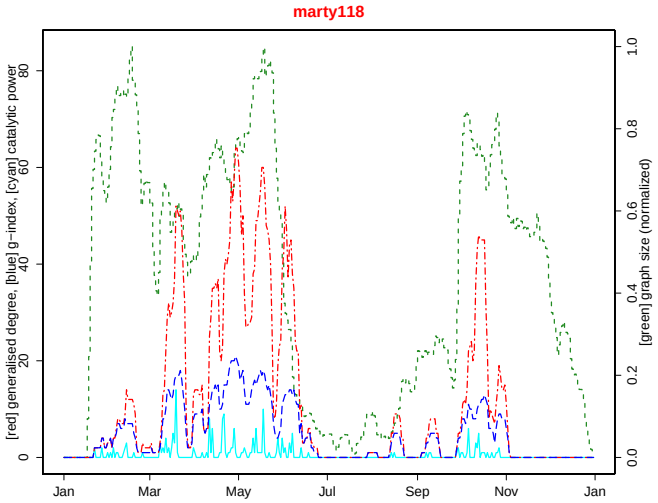


Fig. 1. The activity of a single user over the year, using the windowing method. General forum activity (green dash) is shown, along with g-index (blue long-dash), generalised degree (red dot-dash), and catalytic power (cyan solid).

In order to include this temporal aspect in our analysis, we need to be able to calculate the value of each of the features described in Section 4 for any user at specific moment of time. Seven days appeared to be a natural window for our data, as new episodes of the TV shows in our data set are released on a weekly basis. For every user we calculated a time series of 365 feature vectors, one for each day of the year 2007 observing all interactions within the graph in the past 7 days. Each feature vector contains four values: a user's degree, g-index, catalytic power and cross-topic entropy. The all-zero feature vectors were excluded as they represent moments when users were not part of the community: a user posted a new (non-reply) message, but received no replies to this message. As a result 105,948 feature vectors that belong to 4,291 forum user were retained.

Table 2. Summary of cluster centroids.

Cluster	Generalised Degree	g-Index	Catalytic Power	Cross-Topic Entropy	Number of Observations	Number of # Users
1	0.00	0.00	0.00	5.80	16158	430
2	1.40	3.02	0.00	0.00	22235	848
3	1.39	2.98	3.10	0.00	31294	1879
4	1.64	3.29	3.22	5.73	4867	32
5	2.49	4.17	3.58	0.00	19463	654
6	**4.04**	**5.31**	**4.52**	**0.00**	**6649**	**108**
7	**3.48**	**5.02**	**4.03**	**5.37**	**2501**	**17**
8	1.45	3.07	0.00	5.64	2781	10

5 Experiments

Every user in our collection is now represented by a time-series of four-dimensional daily feature vector observations. Users may take different roles at different times throughout the year, but remain predominantly a single role. We choose to assign the most frequently-occurring and representative role to a user.

In order to surface these roles we must identify the number of possible roles that exist. We perform a k-means clustering (re-initialised 200 times) on the 105,948 vectors created in the previous section. Although k-means is an unsupervised algorithm, it is necessary to provide the initial number of clusters, k, to be created. As we do not make any a priori assumptions on the number of social roles, we have to infer k from the given data.

Out of several existing techniques we chose, $H(k)$, the Hartigan Index. This is a rule-of-thumb proposed by Hartigan, that has been shown to be a highly-effective approach for finding the correct number of clusters [7]. The optimal value for k is the value that maximises $H(k)$. Applying Hartigan's Index (re-initialised 20 times) on the given data set resulted in the optimal number of clusters $k = 8$. For each of our features, x, we performing min-max normalisation (0-128) and transform the value to $log_2 x$ before clustering.

5.1 Clustering Roles

Table 2 presents the cluster centroids along with the numbers of feature vector observations and users within the respective clusters: a number of obvious divisions that have been captured. Cluster 1 contains all observations where a user has posted to many sub-fora (hence the high entropy) but received no replies. Cluster 2 by contrast shows examples of users having replied to well connected (due to the high g-Index and generalised degree) users' posts, but without managing to generate conversation. Cluster 3 is the largest cluster and represents the largest section of the user population who have been reasonably active talking to many different users of no specific prominence. There is a low generalised degree meaning that they do not receive a lot of replies, but they are capable of creating interesting/catalytic posts. Conversely, Cluster 4 contains observations when users create slightly more controversial messages spread across several sub-fora.

Table 3. Top 10 users according to PageRank. Node centrality values are shown for users who appear in the corresponding Top 10 rankings.

User	P.Rank (x10e^{-2})	Degree (x10e^{-1})	Between. (x10e^{-1})	Close.	Cluster (% vectors)
marty118	0.492	**0.455**	0.670	0.340	7 (53.51)
claudiaj	0.420	0.331			6 (76.07)
english toffee	0.373		0.602	0.342	1 (30.02)
Energiya Buran	0.301		0.342	0.323	1 (32.89)
Maybe Once	0.286		0.249		5 (21.71)
Nikki528	0.280	0.345	0.329	0.318	6 (57.89)
Bessie Mae	0.270	0.435	**0.758**	**0.348**	7 (59.90)
jjr	0.263				6 (69.94)
Mozzy55	0.263				3 (64.83)
LollipopGal82	0.262	0.321	0.357	0.332	6 (42.13)

Cluster 7 includes feature vectors with high values of all the features, and so, members of this cluster can be seen as potentially important in several topics. Members of Cluster 6 also have high characteristics in all the features except for entropy. These users may be considered important in a single topic. Note that clusters 6 and 7 contain ∼3% of users: this correlates well with past research [8].

The number of posts create by users follows a standard power-law distribution affecting the role assignment as many of the features are based on activity. It is not directly correlated however as users are only considered during a time-window if they are within the graph. If users write no replies and received no replies (barren threads) they are not within the graph and so these posts do not affect our measures. There is a connection between catalytic power, g-index and message count. This is natural however as the more you post, the more you increase the chance of being noticed. Entropy is not affected by the volume of messages as people interested in a single topic remain so irrespective of posting.

5.2 Validating Social Citation

In order to compare our approach with conventional techniques for identifying important users we calculated degree, betweenness, and closeness centralities as well as the PageRank of each user [9]. Many of the same users returned in the high-valued clusters appear as highly-central nodes in the graph (Table 3). There are some users that seem to be classified very differently by the two approaches. *Mozzy55* for example is the only user from Cluster 3 to make it into the PageRank-based Top 10. PageRank is performed on a static graph containing the entire dataset. *Mozzy55* receives very few replies, but from highly connected users at sporadic intervals throughout the year.

The rule of assigning users to roles using the majority of vectors has some disadvantages. For instance, users “english toffee” and “Energiya Buran” were both assigned to Cluster 1 since this is there most prevalently and accordingly

representative cluster. Upon inspection we saw that both of these users have almost as many occurrences in Cluster 7 respectively. These users appear to be inactive for large proportions of the year, though capable of generating large amounts of conversation when they are active.

6 Conclusions & Future Work

Although our approach helps to identify important users despite inactivity for a period of time, we see that there is important work to be continued on the dynamic evolution/transference from one role to another. In comparing our dynamic approach to that of static PageRank there is inequality. This comparison does however allow us to see that taken as a static measure (max. cluster membership) our measures capture similar features to PageRank. The current work takes the first steps into identifying the roles that users play at specific times allowing the further examination of how these points are joined together: do certain measures rise as others fall? Using a directed rather than undirected network may help distinguish such social roles as "answer people" [1].

In summary we have proposed four measures that aim to capture the interactive style and behaviour of forum users. We have shown that these measures perform well in identifying and clustering users of similar styles, and as a consequence help in the identification of influential users or super-spreaders.

References

1. Welser, H.T., Cosley, D., Kossinets, G., Lin, A., Dokshin, F., Gay, G., Smith, M.: Finding social roles in wikipedia. In: iConference '11: Proceedings of the 2011 iConference, Seattle, WA (2011) 122–129
2. Bakshy, E., Hofman, J., Mason, W., Watts, D.: Identifying 'Influencers' on Twitter. In: WSDM Conference on Web Seach and Data Mining, Hong Kong, China (2011)
3. Forestier, M., Stavrianou, A., Velcin, J., Zighed, D.: Roles in Social Networks: Methodologies and Research Issues. WAIS **10**(1) (2012) 117–133
4. Garfield, E.: Concept of Citation Indexing: A Unique and Innovative Tool For Navigating The Research Literature. (1997)
5. Bornmann, L., Mutz, R., Daniel, H.: Are There Better Indices for Evaluation Purposes Than the h-Index? A Comparison of Nine Different Variants of the h-Index Using Data From Biomedicine. JASIST **59**(5) (2008) 1–8
6. Jamali, S., Rangwala, H.: Digging Digg: Comment Mining, Popularity Prediction, and Social Network Analysis. In: WISM'09: International Conference on Web Information Systems and Mining. (2009) 32–38
7. Chiang, M.M.T., Mirkin, B.: Experiments for the Number of Clusters in K-means. In: Proceedings of the 13th Portuguese Conference on Progress in Artificial Intelligence. EPIA'07, Guimarães, Portugal, Springer-Verlag (2007) 395–405
8. Whittaker, S., Terveen, L., Hill, W., Cherny, L.: The Dynamics of Mass Interaction. In: CSCW'98: Proceedings of the 1998 ACM Conference on Computer Supported Co-operative Work, Seattle, Washington, United States (1998) 257–264
9. Page, L., Brin, S., Motwani, R., Winograd, T.: The Pagerank Citation Ranking: Bringing Order To The Web (1998)