2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining

Early and late fusion methods for the automatic creation of Twitter lists

Mengjiao Wang and Donn Morrison and Conor Hayes
Digital Enterprise Research Institute
National University of Ireland, Galway
Galway, Ireland
Email: firstname.lastname@deri.org

***Abstract*— Twitter's *lists* feature allows users to organize their followees into groups for easier information access and filtering. However, the percentage of users using lists is very small and most existing lists have only a few members. One reason for this may be that curating groups of Twitter users is a time consuming task. In this paper, we propose early and late fusion methods for automatically clustering followees using both graph structure and tweet content. We evaluate our approaches using ground-truth Twitter lists crawled via the Twitter API and show that the late fusion method outperforms both the baselines and the early fusion method.**

***Index Terms*—document clustering; community detection; early fusion; late fusion;**

I. Introduction

The popular Twitter micro-blogging service provides a feature called *lists* enabling a user to organize other Twitter users into groups. The membership information of such lists has proven useful for many data mining and machine learning applications such as curating tweets into meaningful themes [1], finding elite users of a wide range of topics [2], building profiles of user interests [3], and inferring user characteristics and interests [4]. However, as both the list name and list members need to be manually provided by the user, creating and managing Twitter lists is a time consuming task. As a result, the percentage of users using this feature is low. According to our randomly sampled data, less than 4% of Twitter users use lists. We believe that many more users would adopt and use the Twitter list feature if the lists could be created and recommended to them automatically.

In this paper, we introduce methods for automatically creating Twitter lists by clustering followees based on both the followee relations and the content contained in those followees' tweets. In other words, given a set of users, their tweets and their followee relationships, we organize them into several clusters and propose these clusters to the user as a starting point for organizing their followees (Figure 1).

Our primary assumption in this work is that Twitter users create lists to filter information based two main criteria. The first is that a user will want to group his or her followees based on the content or topic of their tweets [4]. For example, a user interested in information retrieval may create a list containing Twitter users who are well known in that community and tweet often about that topic. The second criterion is based on the followee graph. For example, a Twitter users might create a list that contains his or her family members. There is a high likelihood that the members of this list will also follow each other.

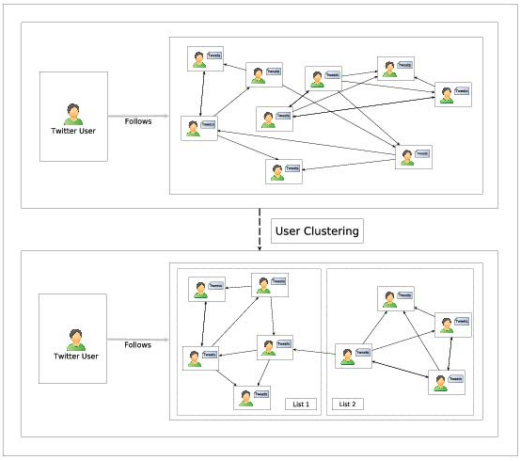

Fig. 1: Clustering a Twitter user's followees for automatically generating curated lists based on tweets and followee graph

To this end, we apply the non-negative matrix factorization (NMF) method based on named entities extracted from tweets and the Order Statistics Local Optimization Method (OSLOM) [5] for community detection based on the followee graph. We then propose early and late fusion methods to combine content features and the followee graph.

The rest of this paper is organized as follows. First, we introduce related work in Section II. Then we describe our Twitter list dataset and baselines in Section III and Section IV. In Section V, we propose our early and late fusion methods. The evaluation methods and experimental results are presented in Section VI. We discuss our experimental results in Section VII, followed by the conclusions and future work in Section VIII.

II. Related work

In the past few years, much research has focused on analyzing Twitter users and the content they produce. Abel et al. [6] investigated hashtag-, entity- and topic-based Twitter user profiles for personalized news recommendations. They

978-0-7695-4799-2/12 $26.00 © 2012 IEEE
DOI 10.1109/ASONAM.2012.203

found that user activity on Twitter was correlated with semantics extracted from news articles. Hannon et al. [7] used tweet content (from the ego and the alters) and collaborative filtering (neighbours' IDs) approaches to create user profiles and recommend Twitter users to follow. They ranked relevant users by a search engine based on a target user profile or relevant query terms.

In the context of Twitter user clustering, there are also some works that study the use of both content and social structure. Zhang et al. [8] investigated the problem of Twitter user clustering based on user interests. They calculated user similarity by linearly aggregating five different user similarity approaches which included similarity of tweet text, URLs, hashtags, followee relationship and retweeting relationship. Karandikar [9] demonstrated a topic model based k-means algorithm for user clustering on Twitter. The author first generated a topic vector which yielded a distribution over each topic for every user. He then used k-means to cluster users based on the topic vector of each user.

More generally, Mei et al. [10] proposed a framework for topic modelling with network structure by assuming that connected documents have similar topic distribution. Their method uses a statistical topic model with a harmonic regularizer learned from network structure and can also be applied to community discovery. Zheng et al. [11] also presented a topic model and network structure-based approach to find topics on participations for use in community discovery. Their method attempts to deal with documents that are usually associated with more than one user (i.e. research paper by multiple authors).

Most of the studies on user clustering have been limited to clustering users/documents into topical groups. In this paper, we investigate the problem of automatically organizing Twitter users into lists according to both topic and structure to encourage the adoption of the Twitter list feature.

III. Preliminaries

A. Crawling Method

In order to have a detailed view of Twitter lists, we crawled data from Twitter via the Twitter APIs. Since Twitter uses an integer number for the user identifier, users were chosen by randomly sampling integers within the range 1 and 500,000,000. Next, we crawled the Twitter lists created by each user. To reduce noise, we ignore lists that have less than five members and users who have less than two lists. For each list, we crawled the list members, the followee relationships between members and the 500 most recent tweets of each list member. While our techniques are not specific to a single language, for the experiments in this paper we focused on English language tweets. To filter non-English people, we used a language detection library[1] and retain only those tweets for which English is the detected language.

[1]Shuyo Nakatani, Language Detection Library for Java: http://code.google.com/p/language-detection/

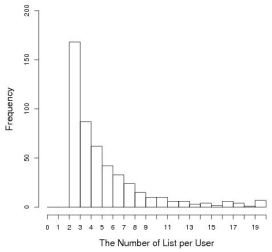

Fig. 2: Histogram of the number of lists per user

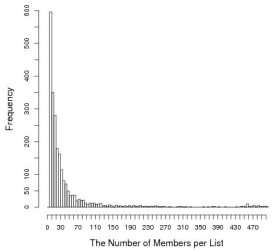

Fig. 3: Histogram of the number of members per list

B. Description

The crawl yielded 490 sample users with a total of 2,311 Twitter lists, 67,083 list members, 2,022,372 followee relationships and 26,694,692 tweets. The number of lists per user and the number of members per list follows an exponential distribution. Most users have less than ten Twitter lists (see Figure 2) and most lists have less than 100 members (see Figure 3). Twitter lists are overlapping sets because Twitter allows users to add a followee to more than one list. In our dataset, 315 users (64%) have overlapping lists.

C. Content Representation

For each tweet we extracted named entities with the UW Twitter NLP Tools[2] which allows for the detection of entities such as persons, geo-location, or products. Then, for each followee of a user U, we use a vector $\vec{v}_U$ to represent the named entities extracted from all of the tweets (to a maximum of 500) sent by this user. Finally, we build a user-term matrix $\mathbf{V}_U \in \mathbb{R}^{M_U \times N_U}$ to represent the user content for the followees of user U, where M_U is the number of followees and N_U is the total number of terms in the resulting vocabulary.

D. Incomplete Information

There are a number of members in Twitter lists that have little or no tweets. In our dataset, there are 3,386 members who have no tweets and there are 7,014 members (10%) have less than 50 tweets. About half of the users (46%) have 500 tweets.

There are also a number of members that do not follow or are not followed by other members in the same list. In other words, these members have no relationships to other members in the list; they are only followed by the users we randomly sampled for crawling. In our dataset, there are 402 (82%) users having at least one such members and also 12 (2%) users have more than 100 such members. Such vertices must be handled as a special case by community detection algorithms as they are unreachable from the rest of the network.

[2]Sam Clark, Alan Ritter, UW Twitter NLP Tools: https://github.com/aritter/twitter_nlp/

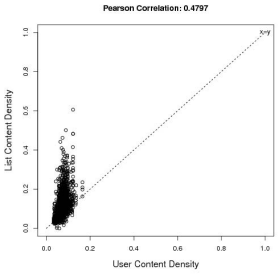

Fig. 4: Content Density

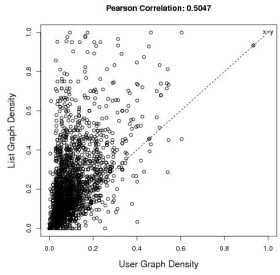

Fig. 5: Graph Density

E. Content Density

To test if followees in the same list share similar content or not, we measure the similarity between followees based on named entities extracted from their content. First we calculate cosine similarity matrix ($\mathbf{S_U}$) from the user-term matrix. Then we define Content Density (**CD**) as the mean cosine similarity of all followees of user U:

$$CD(U) = \frac{\sum_{i \neq i \wedge i,j \subseteq U} S_{ij}}{n(n-1)}. \quad (1)$$

We measure two types of content density: User Content Density and List Content Density. **User Content Density** $CD(U_{all})$ is defined as the content density of all members in all the lists of a sample user and **List Content Density** $CD(U_l)$ is defined the content density of all members in one list of a sample user. Figure 4 shows the plot between User Content Density and List Content Density and suggests that most of the lists share similar content.

F. Graph Density

We posit that members from the same Twitter lists have denser connections. To measure this density, we use the definition of Graph Density (**GD**) for directed graph G:

$$GD(G) = \frac{|E|}{|V_U|\,(|V|-1)}, \quad (2)$$

where $|E|$ is the number of edges in the graph and $|V|$ is the number of nodes in the graph. We also define **User Graph** G_u as all members in sample user's lists and followee relations between and **List Graph** G_l as members only in list l and followee relationships between them. Thus, $GD(G_u)$ is the density of G_u and $GD(G_l)$ is the density of G_l.

For each sample user in our dataset, we calculated the **User Graph Density** and for each list of this sample user, we calculated its **List Graph Density**. Figure 5 shows the plot between User Graph Density and List Graph Density and indicates that most of the lists have denser connections compared to the user graph.

IV. Baselines

Twitter lists are overlapping, meaning members can belong to multiple lists of a user. To address this, we use two baselines: (1) NMF which is a soft clustering method that gives each user (represented by textual content) a latent semantic vector for list membership determination and (2) OSLOM which detects overlapping clusters based on statistical significance of clusters accounting for edge direction and weight.

A. Non-Negative Matrix Factorization

Lee et al. [12] introduced NMF which learns basis vectors from a non-negative co-occurrence matrix. NMF obtains an approximation of $\mathbf{V_U}$ by computing a (W, H) pair to minimize the Euclidean distance of the difference:

$$min\,\|\mathbf{V_U} - WH\| \quad (3)$$

Where $\mathbf{V}_U \in \mathbb{R}^{M_U \times N_U}$ is the user-term matrix. Each dimension of the row vector in matrix W is a base latent topic of the user, and each user is jointly represented by the base latent topics. Non-negative row vectors factorized by NMF can be directly used to find clusters in user collections. Similar techniques like principal component analysis and vector quantization also learn basis vectors, but negative entries in the basis vectors of the latter methods make interpretation difficult. Shahnaz et al. [13] summarized the iterative method to minimize the above function in the following steps:

- Initialize W and H with random non-negative values;
- Iterate for each k, j, and i until convergence or after l iterations:

$$H_{kj} \leftarrow H_{kj} \frac{(W^T\mathbf{V})_{kj}}{(W^TWH)_{kj}} \quad (4)$$

and

$$W_{ik} \leftarrow W_{ik} \frac{(\mathbf{V}^TH)_{ik}}{(W^THH^T)_{ik}}, \quad (5)$$

where k is the number of topics and $0 \leq i \leq M$, $0 \leq j \leq N$.

We use the matrix W to determine the cluster label of each data point. More precisely, examine each row i of matrix W and assign user u_i to cluster k if the k^{th} value in row i is larger than a pre-defined threshold. In this work, we first normalize the Euclidean length of the column vector in matrix W to 1. Then, empirically, we test nine thresholds ranging from 0.1 to 0.9 with a step size of 0.1.

B. OSLOM

Lancichinetti et al. [5] proposed an algorithm called OSLOM that supports finding overlapping community structures in weighted and directed networks. It optimizes locally the statistical significance of clusters with respect to a global null model during community expansion. OSLOM supports overlapping communities, homeless vertices, weighted and directed graph. They are desired because Twitter network are directed and Twitter lists are also overlapping, with some lists sharing followees for a given user. Homeless vertices are those followees that do not belong to any clusters. Although OSLOM allows homeless vertices by default, it can also assign each node to at least one cluster.

V. Early and late fusion

Because we are using information from two sources, namely the followee graph and tweet content, we seek methods that can incorporate both modalities simultaneously and effectively. Information fusion techniques make it possible to combine information from disparate sources with differing representations. Fusion techniques broadly fall into two categories: early and late fusion. *Early fusion* [14], [15] (also called feature fusion) combines features before clustering while *late fusion* [14], [16] (also called model fusion) combines the clustering results generated by different clustering algorithms (see Figure 6).

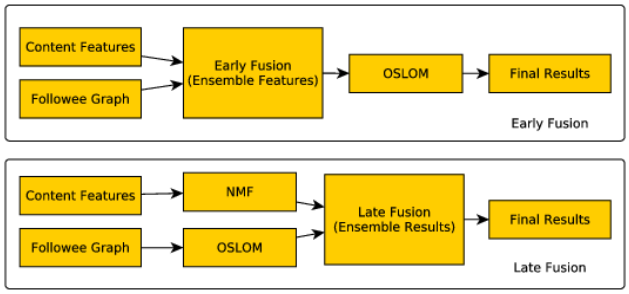

Fig. 6: Early Fusion vs. Late Fusion

A. Early Fusion

The key idea in our early fusion approach is to augment the followee graph for a given user by adding links between followees based on content similarity. We then extract communities from the denser graph. Our early fusion approach is designed as follows:

1) Vectorization: We represent the followee graph for user U as an unweighted adjacency matrix A_U, where each binary element indicates whether there is a connection between two users. We calculate a cosine similarity matrix S_U from the term-document matrix $\mathbf{V}_U$. Each element in S_U is in the interval of $[0, 1]$.

2) Threshold: We set diagonal of matrix S_U to 0, because elements in diagonal are similarity score of users and themselves. To reduce the number of non-zero elements in S_U, we then take largest $p\%$ of elements by sorting the elements in matrix S_U.

3) Edge Weight Assignment: We combine tweet content with the followee graph by assigning edge weights to the unweighted followee graph. Where no followee relationship exists and the textual similarity between two users is greater than a threshold, a new edge is created with the weight equal to the corresponding element in S_U. The steps of edge weight assignment are:

(a) Normalize S_U: set $diag(S_U) = 0$; set $S_U(i, j) = 0$ if $S_U(i, j) < Threshold$
(b) Fusion: define new adjacency matrix $A'_U = A_U + S_U$

4) OSLOM: Finally, we run OSLOM on the new weighted graph, which is represented by adjacency matrix A'_U.

B. Late Fusion

For our late fusion approach, we combine clustering results generated by our two baseline methods with a greedy algorithm. Suppose we have clustering results $\mathbf{C}_1 = \{c_1, c_2, \cdots, c_i\}$ and $\mathbf{C}_2 = \{c_1, c_2, \cdots, c_j\}$ generated by different clustering methods. Our greedy algorithm will merge $\mathbf{C}_1$ into $\mathbf{C}_2$ into a new result $\mathbf{C} = \{c_1, c_2, \cdots, c_k\}$ with the following steps:

For each cluster c_i in $\mathbf{C}_1$
(a) find c_i's most similar cluster c_j in $\mathbf{C}_2$
(b) merge c_i into c_j to get a larger cluster c_k
(c) add c_k to $\mathbf{C}$

We use the Jaccard index to measure cluster similarity which is defined as the size of the intersection divided by the size of the union of the sample sets:

$$J(A, B) = \frac{|A \bigcap B|}{|A \bigcup B|} \tag{6}$$

Two configurations are possible with this greedy algorithm: a) merging NMF results into OSLOM and b) merging OSLOM results into NMF. We investigate both configurations in the following experiments.

VI. Experiments and results

A. Evaluation

1) Normalized Mutual Information: Lancichinetti et al. [17] proposed Normalized Mutual Information (NMI) based on mutual information, which is extended for overlapping sets or *covers*. This measure has become quite popular for comparing community finding algorithms in social network analysis. NMI is defined as:

$$NMI = 1 - \frac{1}{2}\left(\frac{H(X|Y)}{H(X)} + \frac{H(Y|X)}{H(Y)}\right) \tag{7}$$

Where $H(X)$ is the entropy of the overlapping sets X, whereas $H(Y|X)$ is the conditional entropy of X given Y and vice versa. In this paper, we use NMI to measure the similarity between the ground truth lists crawled from Twitter and the clusters found by our method. As such, it yields a mean score for each user U.

$$Mean(NMI) = \frac{\sum_{i=1}^{N} NMI_U}{N} \tag{8}$$

Where NMI_{U_i} is the NMI score of user U and N is the total number of users. In our experiments, N is always 490.

B. Experimental Results

Both NMF and OSLOM are stochastic and rely on randomly initialized values. We run each experiment 5 times with different initial values and calculate mean NMI for each experiment. We also perform one way analysis of variance to ensure our results are significant. We first show the results of the baselines (NMF and OSLOM) followed by the results of early and late fusion. The notation we use for each method is presented in Table I.

Method Notation	Description
NMF-x	the NMF when we use x as threshold
OSLOM-1	the OSLOM with homeless vertices
OSLOM-2	the OSLOM without homeless vertices
Early Fusion-1	the method that runs OSLOM-1 on weighted graph
Early Fusion-2	the method that runs OSLOM-2 on weighted graph
Late Fusion-1	the method that merges NMF-0.8 into OSLOM-1
Late Fusion-2	the method that merges NMF-0.8 into OSLOM-2
Late Fusion-3	the method that merges OSLOM-1 into NMF-0.8
Late Fusion-4	the method that merges OSLOM-2 into NMF-0.8

TABLE I: Description of the notation of different clustering methods

Method	Mean NMI ± Standard Deviation
NMF-0.8 when K=2	0.229 ± 0.0005
OSLOM-1	0.253 ± 0.0018
Early Fusion-1	0.271 ± 0.0024
Late Fusion-2	0.272 ± 0.0011

TABLE II: Overall NMI results of different methods

1) Baselines: From the experimental results in Figure 7, we can see that the method based on the followee graph is better than the method based on textual content. In OSLOM results, the variant that handles homeless vertices performs better than the variant that ignores homeless vertices. For NMF, there is a preference for a smaller number of topics (NMI is higher), indicating that Twitter users tend to track a small number of topical lists.

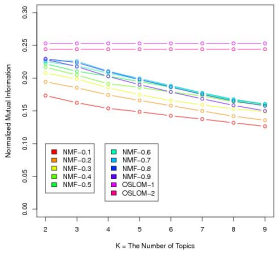

Fig. 7: Baseline Results

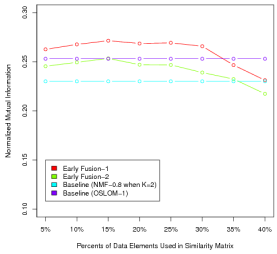

Fig. 8: Early Fusion Results

2) Early Fusion: Figure 8 shows the results of our early fusion approach. In the evaluation of NMI, Early Fusion-1 outperforms Early Fusion-2. Early Fusion-1 reaches the highest NMI value when the percentages of users we take from user similarity matrix is 15%. This indicates that we should introduce a limited amount of content information into the followee graph and it does not necessarily mean that the more content information we introduce, the higher NMI we obtain.

3) Late Fusion: We show the results for our late fusion approach in Figure 9. We can see that the Late Fusion-1 has the highest NMI value when the number of topics is 2. Increasing the number of topics does not yield further improvement in NMI.

4) Overall Results: We compare the NMI results of different methods in Table II. Late fusion slightly outperforms early fusion and both of fusion methods outperform baselines.

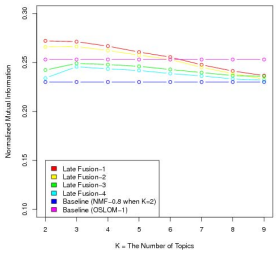

Fig. 9: Late Fusion Results

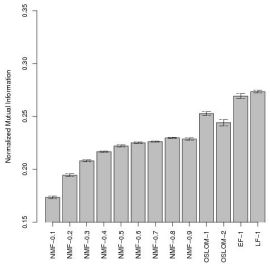

Fig. 10: Mean values and standard deviations

C. Case study

To illustrate the results, we choose a user who has three lists with the first list (red) having 7 members, the second (blue) having 33 members and the third (green) having 9 members. By using the content posted by those members, NMF-0.8 (when the number of topics is 2) finds two overlapping lists. Its NMI score of this user is 0.576. By using the followee relationships, OSLOM-1 also finds two lists for this user and it achieves a NMI score of 0.583. The Early Fusion-1 method combines the followee graph and content information by adding new edges between users with similar content to the followee graph and the Late Fusion-1 method merges the result of NMF-0.8 to the result of OSLOM-1. In this example, Both Early Fusion-1 and Late Fusion-1 improve the clustering results compared the baselines.

VII. Discussion

A. Content vs. Structure

We saw in Figures 4 and 5 that graph density has a higher Pearson correlation value than content density. As a result, the NMI values of OSLOM are better than the NMI values of NMF in our experiments (Figure 7). This indicates that the followee graph is more important than textual content for predicting lists.

B. Stability

To clarify the stability, we plot error bar of mean values and standard deviations of each running of our experiments. Figure 10 shows that the variance of NMI scores of both NMF, OSLOM, Early Fusion-1 and Late Fusion-1 algorithms is low.

Further, to account for variance we perform one-way analysis of variance (ANOVA) of NMI scores on the 5 runs from NMI-0.8, OSLOM-1, Early Fusion-1 and Late Fusion-1. The F-value of this experiment, which is 13.9, means the variance between NMI scores of the algorithms is higher than the variance within NMI scores of the algorithms. The p-value in this experiment is 0.00154, which is smaller than 0.01. The ANOVA shows that the results are statistically significant.

C. Early fusion vs. Late fusion

Both the early fusion and the late fusion methods rely on empirical selection of parameters. Early fusion requires a

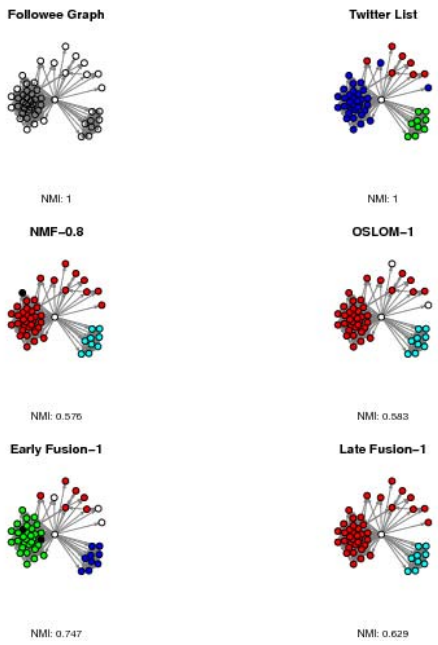

Fig. 11: Example of OSLOM-1, NMF-0.8, Early Fusion-1 and Late Fusion-1 clustering results and NMI score of a user comparing to ground truth lists. The white vertex in the centre is the user for whom we recommends Twitter lists. Vertices in white are homeless which means that our methods fail to assign them to lists. Vertices in red, blue and green are members belong to three lists, Vertices in black are overlapping and belong to more than one list.

threshold to decide how much information from the content similarity matrix to retain before combining it to followee graph. Though late fusion does not rely on any parameters itself, it requires empirical selection of baseline approaches which rely on the number of topics.

VIII. Conclusion and future work

In this paper, we addressed the problem of the tedious manual creation of Twitter lists by clustering tweet content and followee graphs. We proposed several early and late fusion configurations to combine information from both textual content and followee graph. The fusion results show that our proposed approaches have better clustering results than the baseline methods. Although we specifically target the application of Twitter lists, we note that the fusion methods introduced here are generalizable to any domain where content and structure could be combined (e.g., Facebook, Google+).

Future work includes the integration of more features to improve user similarity calculation such as: followed-by, mention-by and retweeted-by, which means that users are similar if they co-followed/co-mention/co-retweeted by the same user. In addition, the current approaches only consider users already in lists. We aim to include all followees and use a user-evaluation rather than the ground truth lists to evaluate our approaches.

IX. Acknowledgements

This work is supported by Science Foundation Ireland (SFI) under Grant No. SFI/08/CE/I1380 (Lion-2 project)

References

[1] D. Greene, F. Reid, G. Sheridan, and P. Cunningham, "Supporting the Curation of Twitter User Lists," *NIPS 2011 Workshop on Computational Social Science and the Wisdom of Crowds*, 2011.

[2] S. Wu, J. Hofman, D. Watts, and W. Mason, "Who Says What to Whom on Twitter Categories and Subject Descriptors," *Proceedings of the 20th International Conference on World Wide Web*, pp. 705–714, 2011.

[3] P. Nasirifard and C. Hayes, "Tadvise : A Twitter Assistant Based on Twitter Lists," *Proceedings of the third International Conference on Social Informatics*, vol. 6984/2011, 2011.

[4] D. Kim, Y. Jo, I.-C. Moon, and A. Oh, "Analysis of Twitter Lists as a Potential Source for Discovering Latent Characteristics of Users," *Workshop on Microblogging at the ACM Conference on Human Factors in Computer Systems*, 2010.

[5] A. Lancichinetti, F. Radicchi, J. J. Ramasco, and S. Fortunato, "Finding Statistically Significant Communities in Networks," *Physcial Review*, vol. 6, no. 4, p. e18961, Jan. 2011.

[6] F. Abel, Q. Gao, G. Houben, and K. Tao, "Analyzing User Modeling on Twitter for Personalized News Recommendations," *User Modeling, Adaption and Personalization*, 2011.

[7] J. Hannon, M. Bennett, and B. Smyth, "Recommending Twitter users to follow using content and collaborative filtering approaches," in *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010, pp. 199–206.

[8] Y. Zhang, Y. Wu, and Q. Yang, "Community Discovery in Twitter Based on User Interests," *Journal of Computational Information*, vol. 3, pp. 991–1000, 2012.

[9] A. Karandikar, "Clustering Short Status Messages: A Topic Model based Approach," *Master Thesis, University of Maryland*, 2010.

[10] Q. Mei, D. Cai, D. Zhang, and C. Zhai, "Topic Modeling with Network Regularization," *Proceeding of the 17th International Conference on World Wide Web*, p. 101, 2008.

[11] G. Zheng, J. Guo, L. Yang, S. Xu, S. Bao, Z. Su, D. Han, and Y. Yu, "Mining Topics on Participations for Community Discovery," *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 445–454, 2011.

[12] D. D. Lee and H. S. Seung, "Algorithms for Non-negative Matrix Factorization," *Advances in Neural Information Processing Systems*, vol. 13, pp. 556–562, 2001.

[13] F. Shahnaz, M. Berry, V. Pauca, and R. Plemmons, "Document Clustering using Non-negative Matrix Factorization," *Information Processing and Management*, vol. 42, no. 2, pp. 373–386, 2006.

[14] C. Snoek, M. Worring, and A. Smeulders, "Early versus Late Fusion in Semantic Video Analysis," *Proceedings of the 13th Annual ACM International Conference on Multimedia*, pp. 399–402, 2005.

[15] Y. Fu, L. Cao, G. Guo, and T. S. Huang, "Multiple Feature Fusion by Subspace Learning," *Proceedings of the 2008 international conference on Content-based image and video retrieval - CIVR '08*, p. 127, 2008.

[16] E. Bruno and S. Marchand-maillet, "Multiview Clustering : A Late Fusion Approach Using Latent Models Categories and Subject Descriptors," in *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009, pp. 736–737.

[17] A. Lancichinetti and S. Fortunato, "Detecting the Overlapping and Hierarchical Community Structure in Complex Networks," *New Journal of Physics*, vol. 11, no. 3, p. 33015, 2009.