A Multi-Resolution Approach to Learning with Overlapping Communities

Lei Tang, Xufei Wang, Huan Liu
Computer Science & Engineering
Arizona State University
Tempe, AZ, USA
{l.tang, xufei.wang, huanliu}@asu.edu

Lei Wang
Engineering & Computer Science
The Australian National University
Canberra, ACT, Australia
lei.wang@anu.edu.au

ABSTRACT

The recent few years have witnessed a rapid surge of participatory web and social media, enabling a new laboratory for studying human relations and collective behavior on an unprecedented scale. In this work, we attempt to harness the predictive power of social connections to determine the preferences or behaviors of individuals such as whether a user supports a certain political view, whether one likes one product, whether he/she would like to vote for a presidential candidate, etc. Since an actor is likely to participate in multiple different communities with each regulating the actor's behavior in varying degrees, and a natural hierarchy might exist between these communities, we propose to zoom into a network at *multiple different resolutions* and determine which communities are informative of a targeted behavior. We develop an efficient algorithm to extract a hierarchy of overlapping communities. Empirical results on several large-scale social media networks demonstrate the superiority of our proposed approach over existing ones without considering the multi-resolution or overlapping property, indicating its highly promising potential in real-world applications.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database applications—*Data Mining*; H.3.3 [**Information Search and Retrieval**]: Clustering

General Terms

Algorithm, Experimentation

Keywords

Multi-Resolution, Overlapping Communities, Hierarchical Clustering, Social Dimensions, Network-based Classification

1. INTRODUCTION

Owing to the rapid development of Internet, emails, participatory Web, social media, and mobile applications, human beings are easier to connect to each other than ever. Millions of users are playing, tagging, working, flirting, and socializing online, producing oceans of data in forms of network interactions each day, opening up a vast range of possibilities to study human relations and collective behavior on an unprecedented scale. In this work, we aim to address the following problem:

> Given a social network and known preferences or behaviors of individuals in the network, how can we employ the connectivity to determine the preferences or behaviors of others in the network?

Here, preferences or behaviors can refer to whether a user supports some political view, whether one likes one product, whether he/she would like to vote for a presidential candidate, etc. This problem is known as within-network classification [18], or a special case of relational learning [12].

1.1 Optimal Number of Communities

In order to resolve this problem, we have to interpret social connections smartly. As suggested by Tang and Liu [26], an actor is likely to be involved in multiple different relations. For instance, a user's online contacts might be categorized by *colleagues*, *classmates*, *relatives*, *researchers*, *co-travelers* and many more. These friends of heterogeneous relations can influence the user in varying degrees. Unfortunately, most social media applications do not provide explicit relation type information. Hence, it is imperative to differentiate these connections presented in a network. The authors suggest that actors involving in the same relationship tend to form a densely-knit group, (say, people of the same class or company are likely to connect to each other). Hence, they propose a *social-dimension* based framework to address the behavior prediction problem. It consists of two steps: the first step extracts communities from a network; the second step builds a classifier by treating each node's community membership as features (which is defined as social dimensions in the work). The framework turns out to outperform other representative approaches based on collective inference.

Though showing promising performance on several social media data sets, the social-dimension framework [26] requires practitioners to specify the number of communities to extract from a network. It seems that a proper parameter can be critical to the success of the framework as indicated in [25]. Consequently, some procedure like cross-validation has to be used to determine a proper parameter. Considering a network with millions of actors, the community extraction can be time-consuming, as each time a new value is set, the whole community extraction procedure has to be restarted again. Thus a natural question is:

Given a social network, can a machine automatically determine the optimal number of communities to extract without cross validation?

The question seems interesting at first glimpse. Indeed, it is often listed as future work in many papers about clustering or community detection. Alternatively, we ask the following question:

Is it ***necessary*** *to determine the optimal number of communities for community detection?*

This question is essentially the underlying motive for us to develop this piece of work. As implied by this work, it is indeed unnecessary to determine an optimal number if a *multi-resolution* approach is adopted.

1.2 Communities of Multiple Resolutions

In reality, actors are involved in multiple relations of which some are associated with a natural hierarchy. For instance, employees working on a small project form a community, which is within a department, which might reside in another larger community representing the whole company. Similarly, the students of a class form a group, which is within the department group. And the department group resides in another group representing the whole university. Evidently, these groups at different resolutions play assorted regulations on one's behavior. Instead of picking a proper parameter, we conjecture that, by *extracting communities at all possible resolutions*, we may be able to avoid a tedious cross-validation procedure.

To find communities of varying resolutions, a natural solution is hierarchical clustering. However, the overlapping nature of different relations complicates the problem. One user is likely to be involved in multiple different communities, and these communities each reside in a hierarchical path. For example, one student might connect to some of his former classmates online. These classmates span in different departments, which in turn form a university-wide group. At the same time, he might connect to his current colleagues, which also reside in a hierarchical structure. In other words, one actor is allowed to reside in multiple different paths in the resultant dendrogram, instead of a single one as commonly studied in existing hierarchical community detection approaches. It requires advanced techniques to find *multi-resolution overlapping communities.*

In this work, we present a learning technique that can extract overlapping communities at different resolutions. We conduct agglomerative hierarchical clustering starting from communities at the finest resolution in order to find out all communities of coarse levels. The key difference of this work from most hierarchical community detection approaches is that it allows one node to reside in multiple different paths in the final dendrogram. We also present a regularization strategy on extracted communities and node affiliations for classifier construction. It turns out the final solution is very simple and effective. The proposed method outperforms baseline methods substantially while requiring no critical parameter input. Meanwhile, the method is efficient, providing a viable solution for practical deployment to handle networks in many applications.

2. RELATED WORK

This work aims to infer user behaviors or preferences given limited labeled nodes and an associated social network. Related work includes both classification with network data and community detection.

2.1 Classification with Network Data

Within-network classification [18, 10] refers to the classification when data instances are presented in a network format. The nodes in social networks, are usually not independent with each other, which differs from conventional data mining or machine learning. To capture the correlation between labels of neighboring data objects, a typical assumption widely adopted is Markov dependency. That is, the labels of one node depend on the labels (or attributes) of its neighbors. Normally, a relational classifier is constructed based on the relational features of labeled data, and then an iterative process is required to determine the class labels for the unlabeled data. The class label or the class membership of each node is updated in turn while the labels of its neighbors are fixed. This process is repeated until the label inconsistency between neighboring nodes is minimized. Such iterative process includes Gibbs sampling [11], iterative classification[16] and relaxation labeling [5]. It is shown that [18] a simple weighted vote relational neighborhood classifier [17] works reasonably well on some benchmark relational data and is recommended as a baseline for comparison. It turns out that this method is closely related to Gaussian field for semi-supervised learning on graphs [32].

A potential limitation with collective inference is that it does not differentiate heterogeneous relations as presented in a network. Typically, people in a social network get connected due to various reasons. These relations correlates with class labels in varying degrees. Collective inference, if applied directly to a social network, does not differentiate these relations. Hence, a social-dimension based framework is proposed [26]. It consists of two steps: extract social dimensions representing different affiliations in a network; and treat social dimensions as features to build classical attribute-based classifier. Since people of the same relation or affiliation form communities, this framework essentially adopt communities extracted from a network as features for classification learning with network data. As an actor is likely to participate in multiple different relations, thus multiple communities, the authors suggest using soft clustering or probabilistic approaches to extract social dimensions. For example, a soft version of modularity optimization is adopted in [24]. However, soft clustering or probabilistic approaches often lead to dense social dimensions, causing computational barrier with large-scale networks. Therefore, the authors define a community as a set of edges and propose EdgeCluster algorithm to extract sparse social dimensions so that the framework can handle mega-scale networks [25]. It essentially partition edges into disjoint sets such that a node can be associated with multiple different communities. In order to partition edges, the authors suggest treating each edge as one instance, and terminal nodes as its features. The resultant EdgeCluster algorithm is reported to handle a network with 1 million nodes in around 10 minutes on a high-end PC. Some other work models the connections using a latent group model [19, 30].

2.2 Community Detection

Finding overlapping communities is attracting increasing attentions. Note that finding overlapping communities is quite a different task from soft clustering [31]. As pointed out in [25], soft clustering often returns a dense community indicator matrix. It destroys the genuine sparsity presented in a network, thus causing many computational problems. On the contrary, the community indicator matrix of overlapping communities is often quite sparse.

Palla et al. propose a clique percolation method to discover overlapping dense communities [21]. It consists of two steps: first find out all the cliques of size k in a graph. Two k-cliques are connected if they share $k-1$ nodes. Based on the connections between cliques, we can find the connected components with respect to k-cliques. Each component then corresponds to one community. Since a node can be involved in multiple different k-cliques, the resultant community structure allows one node to be associated with multiple different communities. A similar idea is presented in [22], in which the authors suggest to find out all the maximal cliques in a network and then perform hierarchical clustering.

On the other hand, Gregory [13] extends the Newman-Girvan method [20] to handle overlapping communities. The original Newman-Girvan method recursively removes edges with highest betweenness until a network is separated into prespecified number of disconnected components. But it only outputs non-overlapping communities. Therefore, Gregory proposes to add one more action (node splitting) besides edge removal. The algorithm recursively splits nodes that are likely to reside in multiple communities into two, or removes edges that seem to bridge two different communities. This process is repeated until the network is disconnected into desired number of communities.

The aforementioned methods enumerate all the possible cliques or shortest paths in a network, whose computational cost is daunting for real-world large-scale networks. Recently, a simple scheme proposed to detect overlapping communities is to define communities as a set of edges, instead of nodes [25, 7, 1]. The network can be converted into an edge-centric view or a line graph and then existing methods to find disjoint communities can be applied. However, this edge-centric view assumes each link is associated with one cluster while weak ties and cross-community connections are common in real-world networks.

Meanwhile, recent techniques are pushing hard for large-scale hierarchical clustering. One commonly used criterion is based on modularity [6], which claims to be as efficient as $O(md \log n)$ where m, n, d are the number of edges, the number of nodes and the depth of resultant dendrogram, respectively. However, it is noticed that the approach has many unbalanced community merge in the initial stage, leading to high computational cost [28]. So the authors suggest biasing the community merge of comparable community sizes. The state-of-the-art method is Louvain method [3]. Its key idea is to greedily expand a community by checking its local neighboring nodes. After a community is formed, the community becomes a super node and the procedure can be applied again. While previous approaches output a binary tree, this method automatically presents a hierarchy that is optimized for modularity. Based on our empirical experience, this method tends to output a very shallow hierarchy, which is not quite suitable for classification.

Hierarchical overlapping communities are also calling for attentions. Ahn et al. [1] apply typical hierarchical clustering to line graph; Lancichinetti et al. [14] rely on iterative local greedy search to expand a seed community based on a fitness function. By changing associated parameters in the fitness functions, communities at different resolutions can be obtained. Please refer to [9] for a comprehensive treatment.

3. ALGORITHM

In this work, we follow the social-dimension based framework proposed in [26] to address classification problem with network data. It consists of two steps:

- Extract communities from a network;
- Treat nodes' community membership as features to build a classifier.

One of the key components in the framework is to extract meaningful communities from a network. As we have mentioned in the introduction, actors are likely to involve in a hierarchy of different communities. These communities at different resolutions might correlate with a class label in varying degrees. Extracting a specified number of communities does not necessarily reflect the hierarchical structure. Therefore, we propose a multi-resolution approach to extract overlapping communities at diverse resolutions.

3.1 MROC: Extraction of Multi-Resolution Overlapping Communities

To extract communities at a variety of resolutions, we here present an agglomerative hierarchical clustering starting from communities at the highest resolution. We emphasize that the method presented here is not the only way to achieve the desired goal. Of course, divisive hierarchical clustering like [13] can also be employed, but its computational cost is prohibitive.

One fundamental question of this multi-resolution approach is, what are the communities at highest resolution in a network? A majority of hierarchical community detection methods [6, 28, 3] start from each node and progressively join two nodes if they are similar. Here, we conjecture that it is a user and his friends (one user's social circle) forming the smallest community.

In many social networks, a high clustering coefficient is observed [4]. Clustering coefficient of one node i is defined as the probability of connections between one node's neighboring nodes [29]. Suppose a node v_i has d_i neighbors, and there are k_i edges among these neighbors, then the clustering coefficient is

$$CC_i = \begin{cases} \frac{k_i}{d_i \times (d_i - 1)/2} & d_i > 1 \\ 0 & d_i = 0 \ or \ 1 \end{cases} \quad (1)$$

where d_i is the degree of node v_i. A high clustering coefficient indicates a set of users interacting with each other frequently. For example, a study on Orkut [2] showed that 78% of interactions happen between users and their friends.

Given these observations, we propose to treat a node and its neighbors as base communities. Note that, this allows one node to participate in multiple different communities. In particular, a node i with degree d_i is associated with up to d_i different communities. Since each user and his friends

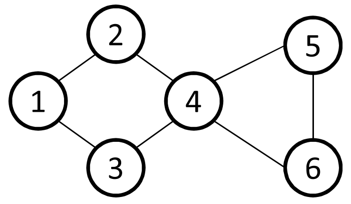

Figure 1: A Toy Example

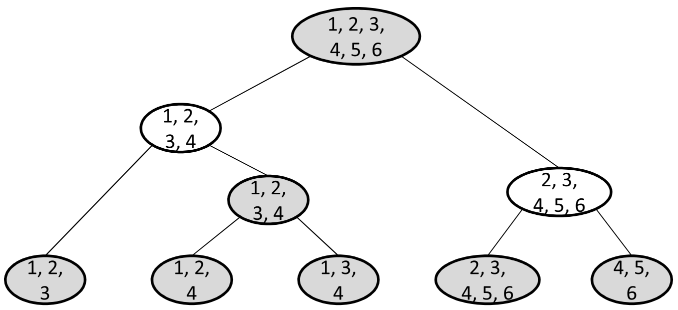

Figure 2: Resultant Dendrogram (shaded ones are newly generated communities)

form a base community, we have at most n base communities in total. For example, in the toy example in Figure 1, there are 6 nodes. The base communities are the 5 communities at the bottom of the dendrogram in Figure 2 (as the base communities constructed from nodes 5 and 6 are the same).

The remaining process is standard: we recursively merge two communities with maximum similarity until all the communities are merged into one or certain criterion is satisfied (say, community sizes exceeding a threshold). Without loss of generality, we use Jaccard index [23] to calibrate the similarity between two communities:

$$J(C_i, C_j) = \frac{|C_i \cap C_j|}{|C_i \cup C_j|} \tag{2}$$

where $|C_i \cap C_j|$ is the number of nodes shared by two communities and $|C_i \cup C_j|$ is the total number of nodes residing in the two communities. For example,

$$J(\{1,2,4\},\{1,3,4\}) = \frac{|\{1,4\}|}{|\{1,2,3,4\}|} = 0.5.$$

Jaccard index has an innate bias toward merging two communities of smaller sizes due to the denominator. Consequently, nodes with high degrees, which would generate a huge base community, would be merged later.

Figure 2 is the resultant dendrogram of applying our proposed multi-resolution approach to the toy example in Figure 1. As shown in the figure, some of the community merges do not produce a new one. Node 4 in the toy example seems to be involved in two different communities $\{1,2,3,4\}$ and $\{4,5,6\}$. Interestingly, both communities are extracted using our approach, but at different resolutions. $\{4,5,6\}$ is the base community produced by node 5, and $\{1,2,3,4\}$ is produced by merging two base communities $\{1,2,4\}$ and $\{1,3,4\}$.

Note that agglomerative hierarchical clustering can take very long time if not implemented in a correct choice. First of all, it is impossible to save similarities between all pairs

```
Input: a network $G(V,E)$;
       maximal community size $\alpha$ for future merge;
Output: social dimensions $\mathcal{S}$.
 1. set $\mathcal{C} = \{\}$;
 2. for each node $i$ in $V$
 3.     append base community $C_i$ into $\mathcal{C}$;
 4.     output the community indicator $\mathcal{S}_{C_i}$;
 5. end
 6. while $\mathcal{C}$ is not empty do
 7.     pop $C_i$ from $\mathcal{C}$;
 8.     $C_j = \arg\max_{C' \in Neighbor(C_i)} J(C', C_i)$;
 9.     $C_{new} = C_i \cup C_j$;
10.     output $\mathcal{S}_{C_{new}}$ if $C_{new} \neq C_i$ and $C_{new} \neq C_j$;
11.     remove $C_i$ and $C_j$ from $\mathcal{C}$;
12.     append $C_{new}$ into $\mathcal{C}$ if $|C_{new}| < \alpha$;
13. end
```

Figure 3: Algorithm for Extraction of MROC

Table 1: Social dimensions of different resolutions

Node	Social Dimensions						
1	1	1	1	0	0	1	1
2	1	1	0	1	0	1	1
3	1	0	1	1	0	1	1
4	0	1	1	1	1	1	1
5	0	0	0	1	1	0	1
6	0	0	0	1	1	0	1

of communities as that would require extensive memory resources. For another, one key observation is that one community can only merge with those communities sharing nodes with it. Hence, it is not necessary to compute all the pairwise similarities between communities. We only need to compute the similarity between one community and its neighboring communities. Further, all the community merges are *local*. Hence, given a set of communities $\mathcal{C} = \{C_i\}$ at certain resolution, we do not need to find the community pair with maximum similarity to merge. Instead, for each community C_i, we locate its neighboring communities and merge C_i with the most similar one (say, C_j). After the merge, C_i and C_j are removed from $\mathcal{C}$ for future consideration. Thus, in one scan of n base communities, we obtain at most $n/2$ newly generated communities. Therefore, starting from $O(n)$ base communities, we only need $O(\log_2 n)$ scans to extract communities at all resolutions, thus saving tremendous computational cost. In practice, we simply maintain a queue for communities. The detailed algorithm is summarized in Figure 3.

In the algorithm, every time we produce a new community, we output its corresponding social dimension representation $\mathcal{S}_C$ (essentially the community indicator vector). For example, the final output for our toy example is shown in Table 1.

3.2 Regularization on Communities

After we obtain the social dimensions representing overlapping communities of various resolutions, we treat them as features and conduct conventional supervised learning. In order to handle large-scale data with high dimensionality and enormous instances, we adopt linear SVM which can be finished in linear time [8]. Generally, the larger a community size is, the weaker the connections inside the community

```
Input:  a network $G(V, E)$;
        labels ($Y^\ell$) for some nodes in the network;
Output: labels of unlabeled nodes ($Y^u$).
1. Extract social dimensions $\mathcal{S}$ of MROC following
   algorithm in Figure 3;
2. Perform regularization on $\mathcal{S}$;
3. Build SVM $\mathcal{M}$ given $\mathcal{S}^\ell$ and $Y^\ell$;
4. Predict labels of unlabeled nodes given $\mathcal{M}$ and $\mathcal{S}^u$.
```

Figure 4: SocioDim Learning with MROC

are. Hence, we would like to build a SVM relying more on communities of smaller sizes. In order to achieve this, we modify typical SVM objective function as follows:

$$\min \lambda \sum_{i=1}^{n} |1 - y_i(\mathbf{x}_i^T \mathbf{w} + b)|_+ + \mathbf{w}^T \Sigma \mathbf{w} \quad (3)$$

where λ is a regularization parameter for SVM[1], $|z|_+ = max(0, z)$ represents SVM hinge loss, Σ is a diagonal matrix to regularize the weights assigned to different communities. In particular, $\Sigma_{jj} = h(|C_j|)$ is the penalty coefficient associated with community C_j. The penalty function h should be monotonically increasing with respect to community size. In other words, a larger weight assigned to a large community results in a higher cost.

Interestingly, with subtle manipulation, the formula in Eq. (3) can be solved using standard SVM package with modified input. Let X represent the input data with each row being an instance (e.g., Table 1), and $\tilde{\mathbf{w}} = \Sigma^{1/2}\mathbf{w}$. Then the formula can be rewritten as

$$\begin{aligned} & \min \lambda \sum_{i} |1 - y_i(\mathbf{x}_i^T \mathbf{w} + b)|_+ + \frac{1}{2}\mathbf{w}^T \Sigma^{\frac{1}{2}} \Sigma^{\frac{1}{2}} \mathbf{w} \\ = \; & \min \lambda \sum_{i} |1 - y_i(\mathbf{x}_i^T \Sigma^{-\frac{1}{2}} \Sigma^{\frac{1}{2}} \mathbf{w} + b)|_+ + \frac{1}{2}\|\tilde{\mathbf{w}}\|_2^2 \\ = \; & \min \lambda \sum_{i} |1 - y_i(\tilde{\mathbf{x}}_i^T \tilde{\mathbf{w}} + b)|_+ + \frac{1}{2}\|\tilde{\mathbf{w}}\|_2^2 \end{aligned}$$

where $\tilde{\mathbf{x}}_i^T = \mathbf{x}_i^T \Sigma^{-\frac{1}{2}}$. Hence, given an input data matrix X, we only need to left multiply X by $\Sigma^{-\frac{1}{2}}$. It is observed that community sizes of a network tend to follow a power law distribution as shown in the experiment part. Hence, we recommend $h(|C_j|) = \log |C_j|$ or $(\log |C_j|)^2$.

On the other hand, one node can be affiliated with different communities. The node affiliations also observe a heavy-tail distribution. Intuitively, a node participating many communities influences less on other group members compared with those "devoted" ones. For effective classification learning, we regularize node affiliations by normalizing each instance's social dimensions to sum up to 1. Later in the experiment, we will study which regularization affects more on classification.

The overall classification learning procedure is summarized in Figure 4. It extracts overlapping communities at multiple resolutions, and then applies certain kind of regularization to feed into SVM learning.

[1]We use a different notation λ from standard formulation to avoid the confusion with community C.

Table 2: Statistics of Social Media Data. Max-Degree denotes the maximum degree, Ave-Degree the average degree, and Clu-Coefficient the average clustering coefficient.

Data Set	BlogCatalog	Flickr	YouTube
Categories	39	195	47
Nodes	10, 312	80, 513	138, 499
Links	333, 983	5, 899, 882	2, 990, 443
Density	6.3×10^{-3}	1.8×10^{-3}	4.6×10^{-6}
Max-Degree	3, 992	5, 706	28, 754
Ave-Degree	65	146	5
Clu-Coefficient	0.49	0.61	0.17

4. EXPERIMENT SETUP

In this section, we present the data collected from social media websites for evaluation, and baseline methods for comparison.

Three benchmark social media data sets in [24] are used to examine our proposed model for collective behavior learning: BlogCatalog[2], Flickr[3] and YouTube[4]. BlogCatalog is a social blog directory where the bloggers can register their blogs under the pre-specified categories. In BlogCatalog, the blogger's interests (categories) are the behavior labels. Flickr and YouTube are both content sharing platforms, with a focus on photos and videos, respectively. In these two sites, users can share their contents, upload tags and subscribe to different interest groups. These interest groups are adopted as users' preference for prediction. The statistics of the data sets are reported in Table 2.

We apply our proposed MROC algorithm to all the three data sets to extract overlapping communities of varying resolutions and convert them to social dimensions for SVM learning with regularization on communities. Since social media networks are very noisy, the connections between group members are weak especially in a large group. Hence, we set α in the algorithm (Figure 3) to 1000. The penalty function of community size is $h = \log |C|$.

Since MROC extracts communities of two properties: overlapping and multi-resolution, two baseline approaches in conjunction with social-dimension framework are included for comparison:

- Learning with disjoint communities in multiple resolutions. We use Louvain method [3] (denoted as Louvain) to extract hierarchical communities from a network. Louvain method is known as the state-of-the-art method in terms of efficiency and cluster quality. It consists of two steps: the first step greedily assigns each node into a cluster such that the modularity is maximized. In the second step a new network is built, with nodes corresponding to a cluster found in the first step, and the weight between two nodes are aggregated from the corresponding clusters. These two steps are repeated until no increase of modularity is possible. Louvain method can only generate non-overlapping clusters in a hierarchy.

- The edge-centric clustering algorithm (EdgeCluster) [25]. EdgeCluster extracts a prespecified number of overlap-

[2]`http://www.blogcatalog.com/`
[3]`http://www.flickr.com`
[4]`http://www.youtube.com`

ping communities from a network. It does not take into account the multi-resolution property associated with communities. The core idea is that while actors are individually involved in multiple overlapping communities, their connections are normally associated with only one community. EdgeCluster views a social network in a different angle by treating edges as instances and nodes as features. It uses an efficient k-means clustering variant to cluster edges into disjoint sets, resulting in overlapping node communities. The resultant social dimensions thus are sparse, enabling the social-dimension based framework to handle mega-scale networks. This algorithm demonstrates comparable classification performances as that based on modularity optimization [24].

For completeness, we also include a recommended collective-inference based method for classification: Weighted-Vote Relational Neighbor Classifier (wvRN) [17]. wvRN has been shown to work well for classification with network data and is recommended as a baseline method for comparison [18]. There is no training necessary and the class labels are determined by the weighted mean of neighbors at prediction. To see the relative significance of performance improvement, we report MAJORITY as well. It always predicts labels based on class prevalence and does not utilize network information at all.

We follow the same evaluation procedure as in [24, 25][5]. Each time, we randomly select part of instances as training data, and the rest as test data. This process is repeated for 10 times for each setup. For SVM training, we use Liblinear [8]. As each data set has more than one categories, the average classification performance in terms of Micro-F1 and Macro-F1 [27] are reported here.

5. EXPERIMENT RESULTS

In this section, we investigate how MROC performs on real-world social media data in terms of classification performance and computational time. Some in-depth analysis of the extracted communities are also reported.

5.1 Prediction Performance Comparison

The performance of various approaches is reported in Tables 3, 4 and 5. The entries in bold denote the best performance. Evidently, social-dimension based approaches (MROC, Louvain, and EdgeCluster) outperform methods based on collective inference. MROC outperforms other methods consistently across all cases for BlogCatalog and Flickr Data. Louvain, though extracts multi-resolution communities, does not allow communities to overlap, thus the predictions are inferior, especially in terms of Macro-F1. EdgeCluster, on the other hand, extracts overlapping communities but does not exhaust communities of various resolutions. Hence, the corresponding learning performance is not comparable to MROC.

Interestingly, EdgeCluster and MROC perform similarly on YouTube data, with MROC being slightly better on Macro-F1 and EdgeCluster on Micro-F1. This is probably because the YouTube data studied is very sparse. Its density is only 4.6×10^{-6} as shown in Table 2. The average degree in YouTube is only 5, way much lower than that of BlogCatalog and Flickr. With such a sparely-connected network, the user interests are very likely to be shared by small-sized communities. Hence, even if we extract communities at higher level of a hierarchy, they cannot help much to infer the preference of actors. Exploring overlapping communities at multiple resolutions is worthwhile, because it improves, in the worse case does not decrease, the performance.

Communities of different resolutions play different roles for behavior prediction. Figure 5 plots the average weights of different communities in the built SVM classifier on BlogCatalog data. The weights in a sense reflect the importance of communities for classification. As seen in the figure, communities of various sizes all contribute to the classification decision function, indicating the necessity to extract communities at multiple resolutions.

[5]`http://www.public.asu.edu/~ltang9/social_dimension.html`

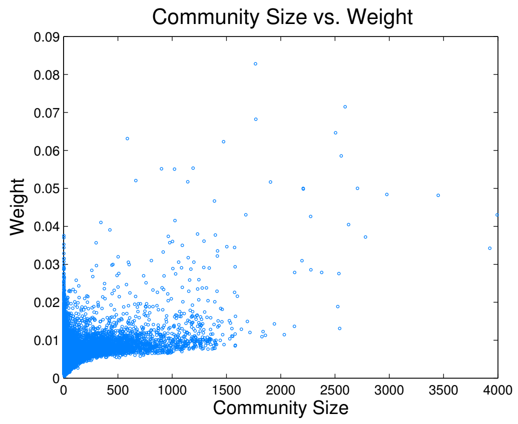

Figure 5: Community Size vs. Relative Importance

Table 6: Computational Time of MROC

Data Set	Time (seconds)	# Communities
BlogCatalog	358	19,773
Flickr	14,714	153, 629
YouTube	9,568	1,034,116

5.2 Time Complexity

The superior performance of MROC comes with higher computational cost. Hierarchical clustering on a network is normally time-consuming. However, by carefully designing the algorithm using local merges as we have introduced in Section 3.1, MROC is able to handle real-world large-scale networks in a reasonable time period. Table 6 shows the computational time (running on an Intel Core2Duo 3.0G CPU) and the number of extracted communities for MROC. For a mega-scale network like YouTube, it takes around 3 hours to finish. On the contrary, Flickr with fewer users (80K) requires around 4 hours to finish. This is because Flickr has many more connections (5.8 million). The neighboring communities considered for a merge multiply in MROC, hence costing more time than YouTube. It is true that MROC takes longer time than EdgeCluster as reported in [25]. Community extraction is a preprocessing step for classification learning in many applications. The classification performance improvement considered, it is often worthwhile to explore MROC if computational time is not critical. In addition, MROC does not require any parameter tuning.

Table 3: Performance on BlogCatalog

Proportion of Training Set		10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1 (%)	MROC	**34.15**	**36.60**	**37.57**	**38.51**	**39.04**	**40.00**	**39.90**	**40.55**	**40.98**
	Louvain	18.17	21.28	23.44	24.65	25.18	25.85	25.71	25.91	26.77
	EdgeCluster	27.94	30.76	31.85	32.99	34.12	35.00	34.63	35.99	36.29
	wvRN	19.51	24.34	25.62	28.82	30.37	31.81	32.19	33.33	34.28
	MAJORITY	16.51	16.66	16.61	16.70	16.91	16.99	16.92	16.49	17.26
Macro-F1 (%)	MROC	**20.43**	**23.49**	**24.80**	**25.93**	**27.05**	**27.49**	**28.30**	**28.25**	**28.59**
	Louvain	9.80	10.75	11.26	11.48	11.60	11.50	11.33	11.46	11.52
	EdgeCluster	16.16	19.16	20.48	22.00	23.00	23.64	23.82	24.61	24.92
	wvRN	6.25	10.13	11.64	14.24	15.86	17.18	17.98	18.86	19.57
	MAJORITY	2.52	2.55	2.52	2.58	2.58	2.63	2.61	2.48	2.62

Table 4: Performance on Flickr

Proportion of Training Set		1%	2%	3%	4%	5%	6%	7%	8%	9%
Micro-F1 (%)	MROC	25.35	**28.97**	**29.74**	**31.67**	**32.51**	**33.34**	**34.07**	**33.89**	**34.44**
	Louvain	22.15	23.12	23.14	23.43	23.54	23.68	23.60	23.55	23.73
	EdgeCluster	**25.75**	28.53	29.14	30.31	30.85	31.53	31.75	31.76	32.19
	wvRN	17.70	14.43	15.72	20.97	19.83	19.42	19.22	21.25	22.51
	MAJORITY	16.34	16.31	16.34	16.46	16.65	16.44	16.38	16.62	16.67
Macro-F1 (%)	MROC	**12.33**	**16.24**	**17.73**	**19.46**	**20.74**	**21.76**	**23.15**	**23.67**	**23.73**
	Louvain	3.94	5.07	4.55	5.22	5.20	5.34	5.57	5.30	5.64
	EdgeCluster	10.52	14.10	15.91	16.72	18.01	18.54	19.54	20.18	20.78
	wvRN	1.53	2.46	2.91	3.47	4.95	5.56	5.82	6.59	8.00
	MAJORITY	0.45	0.44	0.45	0.46	0.47	0.44	0.45	0.47	0.47

Table 5: Performance on YouTube

Proportion of Training Set		1%	2%	3%	4%	5%	6%	7%	8%	9%
Micro-F1 (%)	MROC	**31.87**	**33.96**	35.00	35.30	35.91	36.44	36.51	37.28	38.76
	Louvain	30.21	31.59	29.93	32.50	31.11	27.05	31.56	30.29	30.26
	EdgeCluster	23.90	31.68	**35.53**	**36.76**	**37.81**	**38.63**	**38.94**	**39.46**	**39.92**
	wvRN	26.79	29.18	33.10	32.88	35.76	37.38	38.21	37.35	38.68
	MAJORITY	24.90	24.84	25.25	25.23	25.22	25.33	25.31	25.34	25.38
Macro-F1 (%)	MROC	**24.03**	**26.92**	**28.96**	**29.63**	**30.31**	30.63	**31.42**	**31.64**	**32.84**
	Louvain	18.30	21.08	16.25	21.21	20.66	18.33	20.65	20.28	20.58
	EdgeCluster	19.48	25.01	28.15	29.17	29.82	**30.65**	30.75	31.23	31.45
	wvRN	13.15	15.78	19.66	20.90	23.31	25.43	27.08	26.48	28.33
	MAJORITY	6.12	5.86	6.21	6.10	6.07	6.19	6.17	6.16	6.18

5.3 Distribution and Regularization Effect

In this subsection, we show some distributions of the extracted communities, which in a sense justifies our regularization on communities. The distributions of the size of extracted communities are shown in Figure 6. They all approximately follow a power law distribution: very few communities have abundant members while the majority have a relatively small number of members ranging from tens to hundreds. In a similar vein, the node affiliation distribution also demonstrates a long tail as shown in Figure 7.

If a community is huge, we expect the connections in the group to be weak, thus should weigh relatively less for classification (corresponding to *Community-Size* regularization). Similarly, when one node is associated with many communities, its influence on other group members should be discounted (*Node-Affiliation* Regularization). Table 7 shows the effect of different regularization strategies. Due to space limit, we only present the case of BlogCatalog here. Similar patterns are observed in the other two data sets. *None* means we use boolean representations as in Table 1, *Community Size* divides the entries by the log of the community size, *Node-Affiliation* normalizes each node's community membership indicator to sum to 1, and *Both*, as suggested by the name, considers both the regularization effect of *Community-Size* and *Node-Affiliation*.

As seen in the table, if no regularization is considered, the performance is worst. It seems that the regularization of *Node-Affiliation* affects more on the classification performance than *Community-Size*. After all, the performance is best when both regularization effects are taken. We have to point out that, in this case, it is indeed quite similar to the tf-idf weighting scheme for vector space model of documents. We emphasize that such a weighting scheme for social dimensions can lead to quite a different performance.

6. CONCLUSIONS AND FUTURE WORK

The expanded availability of social media presents many opportunities and challenges to infer collective behavior using social network information. In a social network, actors are often involved in different communities and these communities might form a hierarchy. In this work, we present an efficient and effective approach MROC to extract overlap-

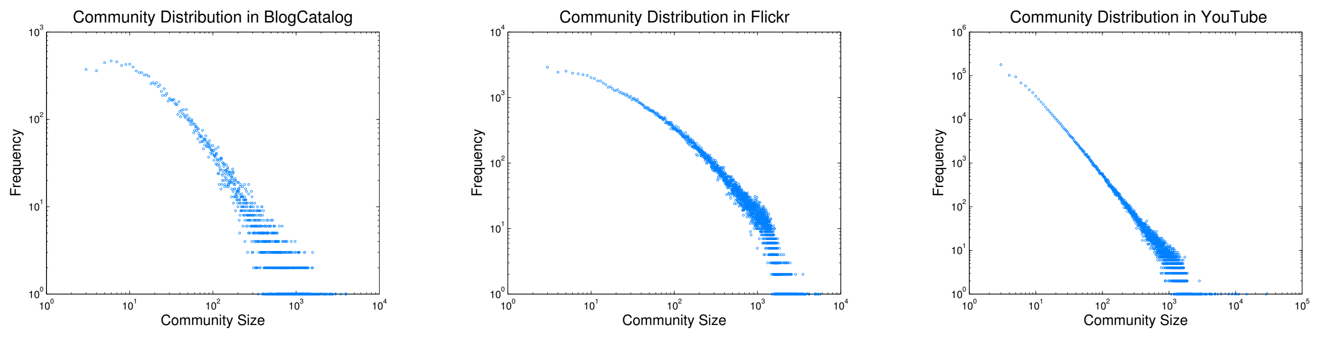

Figure 6: Extracted community size distribution in BlogCatalog, Flickr and Youtube data

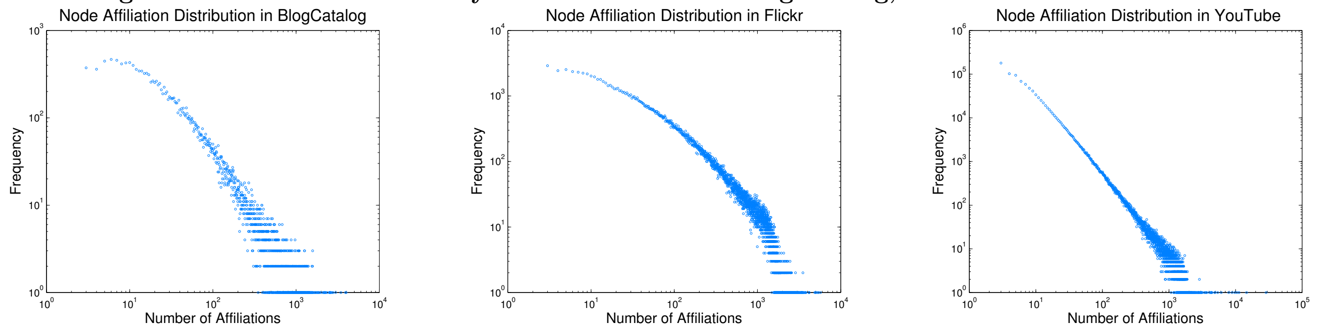

Figure 7: Node affiliation distribution in BlogCatalog, Flickr and Youtube data

Table 7: Regularization Effect on Classification Performance

	Regularization	10%	20%	30%	40%	50%	60%	70%	80%	90%
Micro-F1 (%)	*Both*	**34.15**	**36.60**	**37.57**	**38.51**	**39.04**	**40.00**	**39.90**	**40.55**	**40.98**
	Node-Affiliation	33.50	35.84	36.78	37.93	38.59	39.47	39.06	40.10	40.70
	Community-Size	18.73	24.34	25.62	28.82	30.37	31.81	32.19	33.33	34.28
	None	27.80	29.68	30.87	31.61	32.19	32.99	32.79	34.19	34.09
Macro-F1 (%)	*Both*	**20.43**	**23.49**	**24.80**	**25.93**	**27.05**	**27.49**	**28.30**	**28.25**	**28.59**
	Node-Affiliation	19.82	22.52	24.00	25.19	26.66	26.65	27.28	27.34	28.72
	Community-Size	18.73	20.93	22.62	23.62	24.05	24.66	24.97	25.21	25.54
	None	17.41	19.24	20.57	21.30	22.05	22.55	22.96	23.74	24.40

ping communities at different resolutions, and then utilize them as features for behavior prediction. This in effect does not require users to provide a prespecified number of communities. The learning framework can automatically determine which communities are more relevant to a target behavior. Empirical results on several benchmark social media data sets validate the efficacy of our approach. It suggests that both *overlapping* and *multi-resolution* properties should be considered for community extraction and learning with network data.

Currently, MROC requires hours to extract communities from a network of millions of nodes. It is worthy developing more efficient methods. It is noticed that the optimal spectral cut in large-scale networks often results in communities of size between $100 - 200$ [15]. Does this indicate that communities exceeding certain size does not help for classification? More work can be done to unravel the mystery. We notice that there are some other methods to extract a hierarchy of overlapping communities. A comprehensive comparison of our method with them would be interesting as well.

7. ACKNOWLEDGMENTS

This work is, in part, sponsored by AFOSR and ONR.

8. REFERENCES

[1] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann. Link communities reveal multi-scale complexity in networks, 2009.

[2] F. Benevenuto, T. Rodrigues, M. Cha, and V. Almeida. Characterizing user behavior in online social networks. In *IMC '09: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 49–62, New York, NY, USA, 2009. ACM.

[3] V. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008:P10008, 2008.

[4] D. Chakrabarti and C. Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38(1):2, 2006.

[5] S. Chakrabarti, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *SIGMOD '98: Proceedings of the 1998 ACM SIGMOD international conference on Management of data*, pages 307–318, New York, NY, USA, 1998. ACM.

[6] A. Clauset, M. Newman, and C. Moore. Finding community structure in very large networks. *Arxiv preprint cond-mat/0408187*, 2004.

[7] T. Evans and R. Lambiotte. Line graphs, link partitions, and overlapping communities. *Physical Review E*, 80(1):16105, 2009.

[8] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008.

[9] S. Fortunato. Community detection in graphs. *Physics Reports*, 2009.

[10] B. Gallagher, H. Tong, T. Eliassi-Rad, and C. Faloutsos. Using ghost edges for classification in sparsely labeled networks. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 256–264, New York, NY, USA, 2008. ACM.

[11] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. pages 452–472, 1990.

[12] L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. The MIT Press, 2007.

[13] S. Gregory. An algorithm to find overlapping community structure in networks. In *PKDD*, pages 91–102, 2007.

[14] A. Lancichinetti, S. Fortunato, and J. Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11:033015, 2009.

[15] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 695–704, New York, NY, USA, 2008. ACM.

[16] Q. Lu and L. Getoor. Link-based classification. In *ICML*, 2003.

[17] S. A. Macskassy and F. Provost. A simple relational classifier. In *Proceedings of the Multi-Relational Data Mining Workshop (MRDM) at the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2003.

[18] S. A. Macskassy and F. Provost. Classification in networked data: A toolkit and a univariate case study. *J. Mach. Learn. Res.*, 8:935–983, 2007.

[19] J. Neville and D. Jensen. Leveraging relational autocorrelation with latent group models. In *MRDM '05: Proceedings of the 4th international workshop on Multi-relational mining*, pages 49–55, New York, NY, USA, 2005. ACM.

[20] M. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:026113, 2004.

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

[22] H. Shen, X. Cheng, K. Cai, and M. Hu. Detect overlapping and hierarchical community structure in networks. *Physica A: Statistical Mechanics and its Applications*, 388(8):1706–1712, 2009.

[23] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison Wesley, 2005.

[24] L. Tang and H. Liu. Relational learning via latent social dimensions. In *KDD '09: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 817–826, New York, NY, USA, 2009. ACM.

[25] L. Tang and H. Liu. Scalable learning of collective behavior based on sparse social dimensions. In *CIKM*, pages 1107–1116, New York, NY, USA, 2009. ACM.

[26] L. Tang and H. Liu. Toward collective behavior prediction via social dimension extraction. *IEEE Intelligent Systems*, 2010.

[27] L. Tang, S. Rajan, and V. K. Narayanan. Large scale multi-label classification via metalabeler. In *WWW '09: Proceedings of the 18th international conference on World wide web*, pages 211–220, New York, NY, USA, 2009. ACM.

[28] K. Wakita and T. Tsurumi. Finding community structure in mega-scale social networks: [extended abstract]. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 1275–1276, New York, NY, USA, 2007. ACM.

[29] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

[30] Z. Xu, V. Tresp, S. Yu, and K. Yu. Nonparametric relational learning for social network analysis. In *KDD'2008 Workshop on Social Network Mining and Analysis*, 2008.

[31] K. Yu, S. Yu, and V. Tresp. Soft clsutering on graphs. In *NIPS*, 2005.

[32] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, 2003.