Size Matters: A Comparative Analysis of Community Detection Algorithms

Paul Wagensenell, III, Feng Wang, Member, IEEE, and Weili Wu, Member, IEEE

Abstract—Understanding the community structure of social media is critical due to its broad applications such as friend recommendations, user modeling, and content personalization. Existing research uses structural metrics such as modularity and conductance and functional metrics such as ground truth to measure the quality of the communities discovered by various community detection algorithms, while overlooking a natural and important dimension, community size. Recently, the anthropologist Dunbar suggests that the size of a stable community in social media should be limited to 150, referred to as Dunbar’s number. In this paper, we propose a systematic way of algorithm comparison by orthogonally integrating community size as a new dimension into existing structural metrics for consistently and holistically evaluating the community quality in the social media context. We design a heuristic clique-based algorithm which controls the size and overlap of communities with adjustable parameters and evaluate it along with six state-of-the-art community detection algorithms on both Twitter and DBLP networks. Specifically, we analyze the discovered communities based on their size into four classes called a close friend, a casual friend, acquaintance, and just-a-face, and then calculate the coverage, size of communities with strong ties in both traditional social networks and Internet-based social networks should be limited to 150 (called Dunbar’s number) due to the cognitive constraint and time constraint of human beings. Large communities of size over 150 contain weak connections among their members, therefore, are not stable, while small communities of size 2 or 3 cannot provide the strong sense of team or community. Therefore, we refer to communities of sizes 4–150 as desirable community in this paper and carry out extensive experiments to systematically evaluate and compare various community detection algorithms taking the size of identified communities into consideration.

To make the comparison results reproducible, we adopt six well-known community detection algorithms which have open source implementations, including five disjoint community detection algorithms implemented in igraph package: Infomap [8], Multilevel [9], Fastgreedy [10], Eigenvector [11], and Label Propagation Algorithm (LPA) [12], and one overlapping community detection algorithm called DEMON [13]. In addition, we propose a clique-based algorithm called clique augmentation algorithm (CAA) designed with community size in mind which augments the cliques in the network into communities while using growing threshold and overlapping threshold to control the size of the community and the amount of overlap among communities. We then evaluate these seven algorithms on two data sets with different characteristics, one is a Twitter follower graph and the other is a DBLP network [14]. Specifically, we first study the size distribution

I. INTRODUCTION

COMMUNITY is a natural and fundamental element that exists in a wide variety of networked systems, such as biology, social, and neural networks. Identifying communities in these networks is a crucial step for gaining an in-depth understanding of network structure, dynamics, and interactions. However, there is no unique and widely accepted goodness measure of community quality in literature. Informally, a good community is a densely connected group of nodes, which is sparsely connected to the rest of the network. The extensive research has been devoted to designing community detection algorithms to uncover communities with

Manuscript received January 13, 2018; revised April 28, 2018 and July 22, 2018; accepted September 27, 2018. Date of publication October 30, 2018; date of current version December 3, 2018. This work was supported by the National Science Foundation under Grant 1737861. (Corresponding author: Feng Wang.)

P. Wagensenell, III, and F. Wang are with the School of Mathematical and Natural Sciences, Arizona State University, Glendale, AZ 85306 USA (e-mail: paul.wagensenell@asu.edu; fwang25@asu.edu).

W. Wu is with the Department of Computer Science, University of Texas at Dallas, Richardson, TX 75080 USA (e-mail: weiliwu@utdallas.edu).

Digital Object Identifier 10.1109/TCSS.2018.2875626

IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, VOL. 5, NO. 4, DECEMBER 2018 951

© 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
of the communities identified by each algorithm, then calculate the percentage of users those are assigned to desirable communities for each algorithm. Next, we divide the communities into four different size groups and compare the goodness of communities in each group with the following metrics: 1) extended modularity; 2) TPR; 3) conductance; 4) internal density; and 5) transitivity.

Our experimental results reveal that communities in different groups exhibit diverse structural qualities. We discover that Infomap and CAA outperform others in terms of both community size and coverage by consistently producing desirable communities covering a large portion of users in both Twitter and DBLP networks. Nevertheless, modularity maximization algorithms including Multilevel, Fastgreedy, and Eigenvector and node labeling algorithm LPA tend to output extremely large communities of size over 30,000. Specifically, the largest community identified by Eigenvector holds 71.6% of all users in Twitter graph and 99.5% of all users in the DBLP network, respectively, therefore lacking practical significance.

For modularity, Infomap outperforms others because unlike Multilevel, Fastgreedy, and Eigenvector, whose modularity scores are mostly decided by the large communities of size [501+,] Infomap achieves high score that is contributed by all four groups of its communities. For conductance, modularity maximization algorithms perform better than other algorithm indicating their identified communities are better separated from the external compared to other algorithms. In terms of TPR, DEMON scores 100% in all size ranges due to its algorithm design. CAA outperforms other algorithms when considering TPR reaching above 90% on DBLP network and close to 90% on the Twitter network. All algorithms perform comparably in terms of internal connectivity metrics, transitivity, and internal density, with CAA performing slightly better in these metrics for desirable community sizes.

Our contributions in this paper are twofold:
1) We introduce Dunbar’s number into community detection and evaluation research.
2) We carry out an extensive experiment to systematically evaluate and compare various community detection algorithms taking into consideration the size of identified communities. To the best of our knowledge, this is the first effort to investigate this overlooked but important metric in a systematic way.

The remainder of this paper is organized as follows. Section II describes the importance of size in social networks. Section III introduces the proposed CAA. Performance comparison is given in Section IV. Section V summarizes the related work in community detection algorithm design and evaluation, and Section VI concludes this paper and outlines our future work.

II. COMMUNITY SIZE MATTERS IN SOCIAL MEDIA

Community size is an important factor in social media. On the one hand, groups of sizes 1, 2, and 3 are too small to be called a community. On the other hand, large communities cannot facilitate communication or interaction; therefore, members of the community have limited influence upon each other. Dunbar’s number of 150, which is the limit of stable online community size, is recently introduced to social media [7]. It is calculated as a direct function of relative neocortex size which decides the neocortical processing capacity that limits the number of individuals with whom a stable interpersonal relationship can be maintained, and this in turn limits group size.

Even many online communities have hundreds and even thousands of members, most of those members are inactive, nonparticipatory in the group on a regular basis. Evidence of Dunbar’s number exists ubiquitously, from the size of villages to the size of modern hunter-gatherer societies, to the number of active members in online gaming communities and online forum, and to the number of active administrators in Wikipedia. In addition, a technical forum tends to break down when it reaches about 80 active contributors, requiring a forum split before continued growth could occur [15].

By scrutinizing the top 5000 highest quality communities provided in DBLP, a coauthorship network [14], we found that 4951 of those are of sizes 4–150. This means 99% of the highest quality ground-truth communities in DBLP network conform to Dunbar’s number. Furthermore, the data set claims 13,477 ground truth communities in total and 94% of them have sizes of 4–150. This is another strong evidence of Dunbar’s number.

In addition to the well-known number of 150, Dunbar [7] also provides a series of numbers which correspond to the closeness of a person to people he interacts. According to Dunbar’s model, each person can maintain about five people in their support group of closest friends, about 15 people in a sympathy group who are close enough to confide in, about 50 close friends, about 150 casual friends, and about 500 acquaintances. On the whole, it indicates that any given human can identify about 1500 faces in total. We illustrate these social circles in Table I.

In this paper, seeking to gain a better understanding of how community goodness metrics change with relation to community size, we use these numbers as a guideline to divide the identified communities into different size groups. Specifically, to conform to Dunbar’s breakdown and to avoid overcrowded figures for clear presentation, we group the first six categories in Table I into one group. In this way, the six categories are turned into four classes of communities of size [4–50], [51–150], [151–500], and [501+], respectively. We call these four classes close friend, casual friend, acquaintance, and just-a-face communities.

It is worth noting that we are not stating that communities of large size are not important. Large communities are not practical in social media, but they have their own applications in other types of networks such as biological and neural

| Size  | Closeness    |
|-------|--------------|
| 4–5   | Support clique |
| 6–15  | Sympathy group |
| 16–50 | Close friend   |
| 51–150| Casual friend  |
| 151–500| Acquaintance |
| 501–1500| Just a face  |
networks. Furthermore, we are not reducing the significance of a community only to its size, but it is worth to give in-depth analysis of the quality of communities an algorithm produces by adding community size as an orthogonal dimension to the existing structural measurements.

III. CLIQUE AUGMENTATION ALGORITHM

In this section, we propose a clique based community detection algorithm called CAA. CAA is built on the following two principles: 1) users in a maximal clique belong to a stable community since a clique is densely connected internally and 2) a neighboring node that is highly connected to a clique should be part of the community since it keeps the triadic closure property among all nodes in the community.

Given a social network topology, CAA algorithm discovers communities in the topology using the following steps:

1) Find all maximal cliques in the topology.
2) Filter the overlapping cliques. We sort the cliques based on their size then use an overlapping threshold to control the amount of overlap between two cliques. The overlapping threshold is defined as the percentage of overlapping nodes in the smaller clique. For example, given two cliques $c_1$ and $c_2$, where $c_1$ is of size 10 and $c_2$ is of size 5. Suppose the overlapping threshold is 0.7. If $c_2$ only has two nodes overlapping with $c_1$, we consider $c_1$ and $c_2$ as two independent cliques since $2 < 5 \times 0.7$ which is 3.5. If $c_2$ had four overlapping nodes with $c_1$, we would discard $c_2$ since $4 > 3.5$.
3) Grow each clique into a community by adding new nodes one by one. Growing threshold is utilized for controlling the growth of each community. The growing threshold is defined as the ratio of the number of incoming edges from the new node to other nodes in the community over the size of a community—1. For example, if a community has the size of 10, and the growing threshold is set to 0.7, then for a neighboring node to be added into the community; it must have at least $(10 - 1) \times 0.7 \approx 6$ edges that must come into the intermediate community for a node to be accepted. The algorithm checks the neighboring nodes for each node within the current community and adds node satisfying the growing threshold randomly. This process is repeated for the updated community until no more nodes can be added. The growing threshold allows us to zoomed-in view or out of the graph around the clique. An analysis of the effect of overlapping and growing threshold is carried out in Section IV-C. The implementation detail of the algorithm is presented in Algorithm 1.

The time complexity of CAA is dominated by the time complexity of the maximal clique algorithm, which is NP-Complete. However, consider the moderate clique size in a social network, maximal clique algorithm implemented in NetworkX package runs in polynomial time in practice. After the maximal cliques are obtained, the time to filter overlapping cliques is $O(N^2)$, where $N$ is the number of nodes. The time to augment clique into communities also takes $O(N^2)$.

There are variations of the CAA algorithm based on the sequence of how nodes that satisfy the growing threshold requirement are added to the community. In addition to CAA, which randomly adds nodes to the community, we have experimented with different CAA-based greedy algorithms, including adding a node with maximal incoming edge count, minimum outgoing edge count, and the maximal ratio of incoming over outgoing edge count. We discover that the random algorithm performs the best.

It is worth noting that CAA takes a different approach than the clique percolation method (CPM) where two adjacent cliques are merged into a community structure. In CAA, instead of merging neighboring cliques, we simply grow the community structure by adding an individual node to the community sequentially. CAA has a few nice features: 1) it is faster than CPM and manages to produce similar results and 2) CAA captures the natural growth process of a community in the sense that if a user be friends with many users in a densely connected community, the user will be most likely grow as part of the community.

IV. EMPIRICAL ANALYSIS

A. Description of Data sets

We carry out the comparative research on two data sets: 1) Twitter user follower topology collected over a 3-month period in summer 2013. It contains 318233 Twitter users with 3545258 directed edges. Because not every community detection algorithm we measure in this paper supports the directed graph, we derive an undirected graph by removing all nonmutual edges and the isolated nodes. We call the undirected graph Twitter network in this paper. It contains 190520 nodes and 1001528 undirected edges. 2) DBLP

Algorithm 1 Clique Augmentation Algorithm

1: INPUT: A graph $G$, overlapping threshold $\omega$, growing threshold $\phi$
2: OUTPUT: communityList in graph $G$
3: cliques = findMaximalCliques($G$)
4: filteredCliqueList = filterOverlappingCliques(cliques, $\omega$)
5: foreach clique c in filteredCliqueList:
6:     curCommunity = []
7:     curCommunity.add(c.nodes())
8:     foreach node v in curCommunity:
9:         growingThreshCount = $\lceil((\text{len(curCommunity)} - 1) \times \phi)\rceil$
10:     neighborList = []
11:     foreach neighbor w of v:
12:         if w not in curCommunity && w not in neighborList
13:             add w to neighborList
14:     growingList = []
15:     foreach node u in neighborList:
16:         if w in curCommunity incomingEdgeCount++
17:         if incomingEdgeCount $\geq$ growingThreshCount
18:             add node u to growingList
19:     curCommunity.add(growingList)
20:     communityList.add(curCommunity)
21:     growingThreshCount = growingThreshCount - 1
22: communityList.add(curCommunity)
23: cliques = findMaximalCliques($G$)
24: filteredCliqueList = filterOverlappingCliques(cliques, $\omega$)
25: foreach clique c in filteredCliqueList:
26:     curCommunity = []
27:     curCommunity.add(c.nodes())
28:     foreach node v in curCommunity:
29:         growingThreshCount = $\lceil((\text{len(curCommunity)} - 1) \times \phi)\rceil$
30:     neighborList = []
31:     foreach neighbor w of v:
32:         if w not in curCommunity && w not in neighborList
33:             add w to neighborList
34:     growingList = []
35:     foreach node u in neighborList:
36:         if w in curCommunity incomingEdgeCount++
37:         if incomingEdgeCount $\geq$ growingThreshCount
38:             add node u to growingList
39:     curCommunity.add(growingList)
40:     communityList.add(curCommunity)
41: cliques = findMaximalCliques($G$)
42: filteredCliqueList = filterOverlappingCliques(cliques, $\omega$)
43: foreach clique c in filteredCliqueList:
44:     curCommunity = []
45:     curCommunity.add(c.nodes())
46:     foreach node v in curCommunity:
47:         growingThreshCount = $\lceil((\text{len(curCommunity)} - 1) \times \phi)\rceil$
48:     neighborList = []
49:     foreach neighbor w of v:
50:         if w not in curCommunity && w not in neighborList
51:             add w to neighborList
52:     growingList = []
53:     foreach node u in neighborList:
54:         if w in curCommunity incomingEdgeCount++
55:         if incomingEdgeCount $\geq$ growingThreshCount
56:             add node u to growingList
57:     curCommunity.add(growingList)
58:     communityList.add(curCommunity)
59: cliques = findMaximalCliques($G$)
60: filteredCliqueList = filterOverlappingCliques(cliques, $\omega$)
61: foreach clique c in filteredCliqueList:
62:     curCommunity = []
63:     curCommunity.add(c.nodes())
64:     foreach node v in curCommunity:
65:         growingThreshCount = $\lceil((\text{len(curCommunity)} - 1) \times \phi)\rceil$
66:     neighborList = []
67:     foreach neighbor w of v:
68:         if w not in curCommunity && w not in neighborList
69:             add w to neighborList
70:     growingList = []
71:     foreach node u in neighborList:
72:         if w in curCommunity incomingEdgeCount++
73:         if incomingEdgeCount $\geq$ growingThreshCount
74:             add node u to growingList
75:     curCommunity.add(growingList)
76:     communityList.add(curCommunity)
77: cliques = findMaximalCliques($G$)
78: filteredCliqueList = filterOverlappingCliques(cliques, $\omega$)
79: foreach clique c in filteredCliqueList:
80:     curCommunity = []
81:     curCommunity.add(c.nodes())
82:     foreach node v in curCommunity:
83:         growingThreshCount = $\lceil((\text{len(curCommunity)} - 1) \times \phi)\rceil$
84:     neighborList = []
85:     foreach neighbor w of v:
86:         if w not in curCommunity && w not in neighborList
87:             add w to neighborList
88:     growingList = []
89:     foreach node u in neighborList:
90:         if w in curCommunity incomingEdgeCount++
91:         if incomingEdgeCount $\geq$ growingThreshCount
92:             add node u to growingList
93:     curCommunity.add(growingList)
94:     communityList.add(curCommunity)
data set [14], which is a coauthorship network in computer science where the two authors are connected if they publish at least one paper together. Each node represents an author and each edge indicates coauthorship of a paper. There are totally 317080 nodes and 1049866 undirected edges in DBLP network.

To better understand the characteristic of these two networks, we plot the log–log graph for their node degree distribution in Fig. 1. x-axis is the rank of a node, y-axis is the degree of the node, and a node with a larger degree has a lower rank. It is clear that node degrees in both networks follow power-law distribution. Furthermore, these two networks are different in density and users in a Twitter network have more neighbors than users in DBLP network in general.

### B. Comparison of Community Detection Algorithms

As mentioned in Section I and will be reiterated in Section V, although a lot of research effort has been devoted to the comparison of different community detection algorithms and proposed different evaluation metrics, they ignore a very important factor, the size of the community. In this section, we first study the size distribution of the detected communities by different algorithms and calculate the percentage of users assigned to a desirable community of size between 4 and 150. Then we divide the community by their size into four groups of [4–50], [51–150], [151–500], and [501+] and compare the quality of communities in each group with the following criteria: extended modularity, TPR, conductance, internal density, and transitivity. To make our results reproducible, we adopt the implementation of community detection algorithms in open-access python package igraph and NetworkX, respectively. More specifically, we run Infomap, Multilevel, Fastgreedy, Eigenvector, LPA, Edge Betweenness, WalkTrap, and SpinGlass algorithms in igraph, CPM [16] algorithm in NetworkX, DEMON, and CAA algorithm on both Twitter and DBLP networks. In our experiment, we set the growing threshold of CAA to 0.7, its overlapping threshold to 0, and let communities grow from cliques of size ≥3. It is important to know that setting the overlapping threshold to 0 does not convert CAA to a disjoint algorithm as communities may still grow into each other if they are adjacent. We found that only Infomap, Multilevel, Fastgreedy, Eigenvector, LPA, DEMON, and CAA can handle the scale of the data. Other algorithms hang due to their time complexity and the limitation of the hardware configuration of our experiment environment. Table II summarizes the seven algorithms that finish running. These algorithms can be classified into four categories based on their design ideas as follows: clique based (CAA), network coding and random walk (Infomap), modularity maximization (Multilevel, Fastgreedy, and Eigenvector), and node labeling (Label Propagation and DEMON). In Table II, the overlap indicates whether an algorithm can produce overlapping communities which is more desirable for social media since a user in social media usually belongs to multiple communities. The overlapping community detection is not as extensively studied as disjoint algorithms, this being said we were able to run DEMON and CPM in NetworkX to help compare ourselves to overlapping algorithms. However, we were unable to run CPM on our data sets due to the scale of the data.

1) Community Size Distribution: Tables III and IV summarize the number of communities and the size of the largest community revealed by each algorithm. We also include the percentage of users that are assigned to the largest community. As can be seen, modularity maximization algorithms (Multilevel, Fastgreedy, and Eigenvector) and LPA all produce extremely large communities for both networks. For example, the largest community Eigenvector produces in the Twitter network has a size of 136403, that is, over 70% of all users are grouped into one large community. In the DBLP network, Eigenvector algorithm produces largest community of size 315569, that is, 99.5% of all users are grouped into one large community. The lack of strong connections among community users in such large communities means we can hardly put communities of such large size to practical use.

Tables V and VI provide more detailed breakdown of the number of communities in different size ranges. As can be seen, CAA and Infomap produce more communities in desirable sizes of 4–150 than others. Another interesting observation is that LPA performs significantly better on DBLP network than on the Twitter network. We also notice that DEMON outputs the most communities of size above 150+ among all algorithms. Eigenvector performs poorly on both networks, especially on DBLP network since all it discovers are two large communities of size greater than 501. From the community size distribution point of view, we would recommend CAA and Infomap that can produce a decent number of communities with desirable sizes.
TABLE II
Summary of Algorithms

| Name          | Overlap | Time Complexity                  | Category               |
|---------------|---------|----------------------------------|------------------------|
| CAA           | Yes     | Exponential, fast in practice for small cliques | Clique based           |
| DEMON         | No      | \(O(N + E)\)                      | Vote based node labeling |
| Infomap       | No      | \(O(E)\)                          | Network coding and random walk |
| Multilevel    | No      | \(O(N \log N)\)                   | Modularity maximization |
| Eigenvector   | No      | \(O(N(E + N))\)                   | Modularity maximization |
| Fastgreedy    | No      | \(O(N \log^2(N))\)                | Modularity maximization |
| Label Propagation | No | \(O(E)\)                      | Node labeling |

TABLE IV
Total Number of Detected Communities and the Size of the Largest Community in DBLP Network

| Algorithm | Number Of Communities | Largest Community Size / Ratio |
|-----------|-----------------------|-------------------------------|
| CAA       | 28,213                | 270 / 0.085%                 |
| DEMON     | 8,900                 | 26,830 / 8.5%                |
| Infomap   | 16,599                | 587 / 0.0185%                |
| Multilevel | 565                | 30,427 / 9.6%                |
| Eigenvector | 2                  | 315,569 / 99.5%              |
| Fastgreedy | 3,206                | 54,783 / 17.28%              |
| Label Propagation | 21,156          | 84,363 / 56.6%               |

TABLE V
Community Size Distribution in Twitter Network

| Size Range | CAA | DEMON | Infomap | Multilevel | Eigenvector | Fastgreedy | Label Propagation |
|------------|-----|-------|---------|------------|-------------|------------|-------------------|
| 1 - 3      | 381 | 341   | 641     | 513        | 586         | 323        | 625               |
| 4 - 30     | 3212 | 2732  | 4429    | 3783       | 3863        | 2218       | 2826              |
| 31 - 120   | 454 | 454   | 751     | 952        | 1049        | 551        | 649               |
| 121 - 300  | 500 | 500   | 802     | 1002       | 1102        | 602        | 700               |
| 301+       | 2   | 2     | 25     | 40         | 50         | 15         | 25                |

TABLE VI
Community Size Distribution in DBLP Network

| Size Range | CAA | DEMON | Infomap | Multilevel | Eigenvector | Fastgreedy | Label Propagation |
|------------|-----|-------|---------|------------|-------------|------------|-------------------|
| 1 - 3      | 398 | 398   | 497     | 497        | 502         | 302        | 548               |
| 4 - 30     | 3230 | 2730  | 4423    | 3783       | 3863        | 2218       | 2826              |
| 31 - 120   | 454 | 454   | 751     | 952        | 1049        | 551        | 649               |
| 121 - 300  | 500 | 500   | 802     | 1002       | 1102        | 602        | 700               |
| 301+       | 2   | 2     | 25     | 40         | 50         | 15         | 25                |

2) Community Coverage: In this section, we define a new metric called desirable community coverage to measure the number of users assigned to desirable communities, that is, communities of size 4–150. Generally speaking, the higher the desirable community coverage, the better the algorithm. Note that even though Infomap, Multilevel, Eigenvector, and Fastgreedy all assign every single node in the graph to a community, they do not provide 100% coverage in our definition of desirable community coverage. Fig. 2 shows the performance of each algorithm with regard to desirable community coverage. It is clear that Infomap performs the best in terms of desirable community coverage since it assigns 62% of all users in the Twitter network and 98% of all users in DBLP network into desirable communities. CAA and DEMON are comparably well in the DBLP network and CAA outperforms DEMON in the Twitter network. It is worth noting that in this experiment, CAA starts from nonoverlapping cliques by setting the overlapping threshold to 0. In fact, CAA can achieve higher coverage if we allow overlapping cliques to start with. LPA performs inconsistently in two networks and Fastgreedy, Multilevel, and Eigenvector failed to assign the majority of users in the network to meaningful communities.

3) Extended Modularity: It has been defined as a popular metric to measure the goodness of overlapping communities in [32]. We define this metric in (1), where \(C_i\) is the \(i\)th community, \(O_v\) is the number of communities the vertex \(v\) belongs to, similarly \(O_w\) is the number of communities vertex \(w\) belongs to. \(A\) is the adjacency matrix, that is, \(A_{vw} = 1\) means there exists an edge between vertex \(v\) and vertex \(w\), otherwise it is 0. \((k_v k_w / 2m)\) describes the expected number of edges between vertex \(v\) and vertex \(w\), and \(m\) is the total number of edges in the whole topology. The range of extended modularity is \([-1, 1]\) and the higher the value, the better the community in terms of modularity. The intuition behind extended modularity is that communities should have more internal connectivity than random graph of the same degree sequence. Extended modularity is a variation to modularity, which is the most widely used metric to measure the goodness of disjointed community. If there is no overlap between the communities, \((1/O_v O_w)\) is 1 and (1) becomes the traditional modularity as defined in [17]. In general, overlapping communities have lower modularity than disjoint communities since overlapping communities have many connections outside the community.

\[
EQ = \frac{1}{2m} \sum_{i \in C} \left[ \sum_{v \in C_i, w \in C_j \cap C_i} \frac{1}{O_v O_w} \left( A_{vw} - \frac{k_v k_w}{2m} \right) \right].
\]

As shown in (1), each community contributes a value toward the modularity score. To measure what degree communities of different sizes impact the modularity, we divide the sum in (1) into four parts, that is, the sum over all communities...
of size [4–50], [51–150], [151–500], and [501+] and present the partial modularity score contributed by each community group in Fig. 3. Infomap and DEMON outperform others because unlike Multilevel, Fastgreedy, and Eigenvector, whose modularity scores are mostly decided by the large communities of size [501+], Infomap and DEMON achieve high score that is contributed by all four groups of its communities. Infomap’s modularity scores are very close to the highest score achieved by Multilevel. LPA performed inconsistently in the [4–50] range when comparing the two data sets. The observation that larger communities achieve higher modularity score is consistent with the resolution limit of modularity as indicated in [33]. This discovery also suggests we should maximize modularity while controlling the community size at the same time. It is also worth noting that, in general, overlapping communities have lower modularity than nonoverlapping ones thus the low modularity of CAA is the expected behavior.

4) Triangle Participation Ratio: It was proposed in [6] as a metric for community evaluation, where it is defined as the number of nodes in a community that forms a triad, divided by the total number of nodes in the community. Reference [6] found that the communities discovered by algorithms to optimize community modularity do not align with the ground truth communities, that is, the known membership in real life, while TPR is a good metric when looking for ground-truth communities. As indicated in Fig. 4, DEMON scores a perfect value of 1.0 in all categories due to its design. Nevertheless, CAA achieves significantly higher TPR than others on both the networks, with close to 0.9 TPR score on the Twitter network and above 0.9 TPR score on DBLP network. This is expected since CAA starts from a clique with TPR as 1.0 and grows the clique with new nodes that are highly connected to the growing community. We can also see that all algorithms show satisfactory performance on DBLP network in terms of TPR.

5) Conductance: It is a metric that considers both external and internal connections of a community [5]. It is defined as the ratio of the number of edges on the boundary of the community over the sum of degrees of nodes in the community. Lower conductance score indicates a better community. We calculate the average conductance of communities in different size groups, shown in Fig. 5. We find that Multilevel achieves the lowest average conductance score and all three modularity maximization algorithms perform well. Both CAA and DEMON perform poorly in conductance when compared to modularity maximization algorithms, which is expected since they produce overlapping communities.

6) Internal Density: Internal density [5] is defined in (2), where $m_S$ is the number of edges in the subset $S$ divided by the total possible edges between all nodes $n_S(n_S - 1)/2$. Internal density is a measurement of the internal structure within the community $f(S) = \frac{m_S}{n_S(n_S - 1)/2}$. (2)

Fig. 6 shows the average internal density for communities in each size group. We find that smaller community size typically indicates higher internal density value. This is not surprising as communities with smaller sizes tend to form cliques. All algorithms show similar performance and CAA consistently perform slightly better across all size groups on the DBLP network.

7) Transitivity: It [18] measures the ties between individuals and is defined in (3). This value indicates the probability that two friends of a single person are also friends. A clique has transitivity of 1.0. Higher transitivity value is better since it indicates that the community is more tightly connected to one another and has a higher probability that users within the community know each other

$$\frac{3 \times \text{number of triangles}}{\text{number of triads}}.$$ (3)

The result of the average transitivity score is presented in Fig. 7. We notice that CAA and DEMON perform similar in DBLP network, and DEMON performs slightly better in Twitter network. All algorithms perform similarly in the range [51–150].

In summary, the experimental results suggest that Infomap and CAA outperform others in terms of both community size and coverage by consistently producing desirable communities covering a large portion of users in both Twitter and DBLP networks. Nevertheless, modularity maximization algorithms including Multilevel, Fastgreedy, and Eigenvector and node labeling algorithms including Label Propagation (LPA) and DEMON tend to output extremely large communities. In terms of modularity, Infomap performs the best in general. The
identified community of different sizes contributes to the modularity score and its total modularity is comparable to modularity maximization algorithms. For conductance, modularity maximization algorithms perform better than other algorithms indicating that their identified communities are better separated from the external compared to other algorithms. Regarding TPR, DEMON achieves 100% and CAA outperforms others reaching above 90% in DBLP network and close to 90% in Twitter network. All algorithms perform similarly in terms of internal connectivity metrics, transitivity, and internal density, while CAA performs slightly better for desirable community sizes.

C. Growing Threshold and Overlapping Threshold

In this section, we measure the impact of growing threshold and overlapping threshold on community size and the number of communities in order to give suggestions on the parameter selection for CAA algorithm.

First, we investigate the effect of growing threshold. To do this, we find all cliques of size 3 and larger in DBLP network and close to 90% in Twitter network. All algorithms perform similarly in terms of internal connectivity metrics, transitivity, and internal density, while CAA performs slightly better for desirable community sizes.
join the community. The growing thresholds of 0 and 1 have been omitted because at 0 the result would be the topology surrounding the clique, instead if this is set to 1 the result is the cliques. The result is plotted in Fig. 8, where $x$-axis is the growing threshold for different values and $y$-axis is the number of communities that fall in different size ranges. As can be seen, as growing threshold increases, the number of larger communities of size $[51–150]$, $[151–500]$, and $[501+]$ decreases while the number of smaller communities of size $[4–50]$ increases with an exception at growing thresholds 0.8 and 0.9. This is expected since the smaller the growing threshold, the lower the requirement for a node to join a community so the community tends to grow larger. The same is true for the larger growing threshold since it is harder for a node to join the community as growing threshold increases, therefore more communities with smaller size exist. The exception at growing thresholds 0.8 and 0.9 is due to the rounding of a floating-point value. To best illustrate what is happening, consider a clique of size 3 and the growing threshold is set to 0.7 versus 0.8. When the growing threshold is 0.7 where $(3 - 1) \times 0.7$ equal to 1.4, due to the rounding of the floating point, a node only needs to connect to one node in this clique of size 3 in order to add it to the community. While for growing threshold 0.8 where $(3 - 1) \times 0.8$ equals to 1.6 and is rounded to 2, a node needs to connect to two users in the clique of size 3 to join. Therefore, for growing thresholds 0.8 and 0.9, many cliques of size 3 do not grow resulting in a smaller number of communities than growing threshold 0.7. In general, we recommend to choose growing threshold greater or equal to 0.5 to avoid large community size.

Next, we investigate the effect of overlapping threshold. We choose all cliques of size over 15 and increase the overlapping threshold from 0 to 1. Intuitively, by increasing the overlapping threshold, fewer cliques are filtered; therefore, the number of communities increases. As shown in Fig. 8, where $x$-axis is the overlapping threshold value, $y$-axis is the number of cliques, the number of cliques increases significantly for the overlapping threshold $\geq 0.8$. In general, we suggest choosing an overlapping threshold less than 0.6 to avoid having heavily overlapping communities.

V. RELATED WORK

A lot of effort has been devoted to the area of community detection along with ways to determine the quality of the identified communities. Existing community detection algorithms can be categorized into disjoint algorithms and overlapping algorithms, based on whether the identified communities have overlap or not. Infomap [8] stands out as the most popular and widely used disjoint algorithm. It is based on random walks on networks combined with coding theory with the intent of understanding how information flows within a network. Multilevel [9] is a heuristic based algorithm based on modularity optimization. Multilevel first assigns every node
to a separate community, then selects a node and checks the neighboring nodes attempting to group the neighboring node with the selected node into a community if the grouping results in an increase in modularity. Newman’s Leading Eigenvector [11] works by moving the maximization process to the eigenspectrum to maximize modularity by using a matrix known as the modularity matrix. Fast Greedy [10] is based upon modularity as well. It uses a greedy approach to optimize modularity. LPA [12] works by assigning a unique label to each node in the graph. The nodes are then listed in a random sequential order which the algorithm follows to diffuse the labels through the network. This causes nearby neighboring nodes to adopt the same label from the nodes nearest to them. This process causes a community like sets of nodes to quickly converge to a final label that uniquely identifies the group. TSP–CDA [19] uses existing solutions to the traveling salesman problem combined with a PageRank distance metric to reveal the network community structure. TSP–CDA determines the cuts on the graph using a threshold on the PageRank distance metric allowing for proper community identification from the traveling salesman process. Agent-based genetic algorithms have also been proposed such as MAGA-Net [20], which uses a lattice structure for the agents which can then communicate and use genetic operators to maximize the modularity. This process allows MAGA-Net to reveal community structure while not getting trapped in local optima.

In the category of the overlapping algorithm, CPM [16] merges two cliques into a community if they overlap more than a threshold. The DEMON algorithm [13], [21] lets every node within the network to vote for the communities present locally. This is done by analyzing the ego network of each node and then running the label propagation algorithm on the structure ignoring the ego itself. Another relevant paper [22] discusses overlapping community detection algorithms along with various quality measurements.

One comprehensive survey of recent advances [1] discusses a wide range of existing algorithms including traditional methods, modularity-based methods, spectral algorithms, dynamic algorithms, and more. Community detection often tries to optimize various metrics such as modularity as described by Girvan and Newman [23] or conductance. The study in [5] discusses many of the various objective functions currently in use and how they perform. Similarly, reference [6] points out that it is important for the community detection algorithm to extract functional communities based on ground truth, where the functional ground-truth community is described as a community in which an overall theme exists. Another recent paper [3] conducts a comparative analysis of community detection algorithms on the LFR benchmark in order to provide unbiased guidelines to choose algorithms for a given network. They propose a method to quantify the accuracy of the algorithm with mixing parameters. In addition, Wang et al. [2] provide a procedure-oriented framework for benchmarking and compare different algorithms. The authors also fit and implement 10 algorithms in their framework and compare and rate the algorithms. While we cover analysis of algorithms on single-layer networks, multiplex networks [24] and community detection algorithms on them [25], [26] are another important which often overlooked the area of research. Signed networks in which links may be negative are also an important area of community detection research, especially when it comes to social networks. References [27] and [28] present a few different ways to perform community detection on signed networks using some evolutionary and memetic algorithms to find community structure.

Our work is different from existing comparative research in which we propose a new dimension of comparing the structural metrics for community detection algorithms by considering the size of the identified communities. Our work is the first to investigate a previously overlooked community size in a systematic way.

VI. Conclusion

In this paper, we carry out a comparative analysis of community detection algorithms by introducing community size as a new orthogonal dimension to existing structural metrics. In order to make sense of communities of different sizes, we introduce the Dunbar’s number in sociology into the community detection in social media. We find that Infomap and CAA are able to discover the communities of desirable sizes. We also present and compare the performance of each evaluated algorithm in all evaluated structural metrics.

Our future work includes exploring a greedy approach to improve the performance of CAA in terms of conductance and modularity, conducting a comparison with more overlapping community detection algorithms. Furthermore, we plan to explore new metrics such as whether and in what degree a community interest exists in the identified community for community quality evaluation and also make sense of the community.

REFERENCES

[1] S. Fortunato, “Community detection in graphs,” Phys. Rep., vol. 486, nos. 3–5, pp. 75–174, 2010.
[2] M. Wang, C. Wang, J. X. Yu, and J. Zhang, “Community detection in social networks: An in-depth benchmarking study with a procedure-oriented framework,” in Proc. 41st Int. Conf. VLDB, 2015, pp. 998–1009.
[3] Z. Yang, R. Algesheimer, and C. J. Tessone, “A comparative analysis of community detection algorithms on artificial networks,” Sci. Rep., vol. 6, Aug. 2016, Art. no. 30750.
[4] S. Harenberg et al., “Community detection in large-scale networks: A survey and empirical evaluation,” WIREs Comput. Stat., vol. 6, no. 6, pp. 426–439, 2014.
[5] J. Leskovec, K. J. Lang, and M. Mahoney, “Empirical comparison of algorithms for network community detection,” in Proc. 19th Int. Conf. World Wide Web, 2010, pp. 631–640.
[6] J. Yang and J. Leskovec, “Defining and evaluating network communities based on ground-truth,” in Proc. ICDM, Dec. 2012, pp. 745–754.
[7] R. I. M. Dunbar, “Do online social media cut through the constraints that limit the size of offline social networks?” Rsoc. Open Sci., vol. 3, no. 1, p. 150292, Jan. 2016, doi: 10.1098/rsos.150292.
[8] M. Rosvall and C. T. Bergstrom, “Maps of random walks on complex networks reveal community structure,” Proc. Nat. Acad. Sci. USA, vol. 105, no. 2, pp. 1118–1123, 2008.
[9] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” J. Stat. Mech., Theory Exp., vol. 10, p. P10008, Oct. 2008.
[10] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top., vol. 69, no. 6, p. 066133, 2004.
[11] M. E. J. Newman, “Finding community structure in networks using the eigenvectors of matrices,” Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top., vol. 74, no. 3, p. 036104, 2006.
[12] U. N. Raghavan, R. Albert, and S. Kumara, “Near linear time algorithm to detect community structures in large-scale networks,” Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top., vol. 76, no. 3, p. 036106, 2007.
[13] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi, “Uncovering hierarchical and overlapping communities with a local-first approach,” ACM Trans. Knowl. Discovery Data, vol. 9, no. 1, p. 6, 2014.
[14] J. Yang, and J. Leskovec, SNAP Network Datasets: DBLP Collaboration Network. Accessed: Aug. 2017. [Online]. Available: http://snap.stanford.edu/data/com-DBLP.html
[15] M. Konnikova. The Limits of Friendship. Accessed: Aug. 2017. [Online]. Available: http://www.newyorker.com/science/maria-konnikova/social-media-affect-math-dunbar-number-friendships
[16] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, “Uncovering the overlapping community structure of complex networks in nature and society,” Nature, vol. 435, no. 7043, pp. 814–818, 2005.
[17] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top., vol. 70, no. 6, p. 066111, 2004.
[18] M. Girvan and M. E. J. Newman, “Community structure in social and biological networks,” Proc. Nat. Acad. Sci. USA, vol. 99, no. 12, pp. 7821–7826, 2002.
[19] Z. Jiang, J. Liu, and S. Wang, “Traveling salesmen problems with pagerank distance on complex networks reveal community structure,” Phys. A, Stat. Mech. Appl., vol. 463, pp. 293–302, Dec. 2016.
[20] Z. Li and J. Liu, “A multi-agent genetic algorithm for community detection in complex networks,” Phys. A, Stat. Mech. Appl., vol. 449, pp. 336–347, May 2016.
[21] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi, “DEMON: A local-first discovery method for overlapping communities,” in Proc. KDD, 2012, pp. 615–623.
[22] J. Xin, S. Kelley, and B. K. Szymanski, “Overlapping community detection in networks: The state-of-the-art and comparative study,” ACM Comput. Surv., vol. 45, no. 4, p. 43, 2013.
[23] M. E. J. Newman and M. Girvan, “Finding and evaluating community structure in networks,” Proc. Nat. Acad. Sci. USA, vol. 99, pp. 3282–3286, Feb. 2004.
[24] M. Kivelä A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter. “Multilayer networks.” J. Complex Netw., vol. 2, no. 3, pp. 203–271, Jul. 2014, doi: 10.1093/comnet/cnu016.
[25] X. Wang and J. Liu, “A layer reduction based community detection algorithm on multiplex networks,” Phys. A, Stat. Mech. Appl., vol. 471, pp. 244–252, Apr. 2017.
[26] A. Tagarelli, A. Amelio, and F. Gullo, “Ensemble-based community detection in multilayer networks.” Data Mining Knowl. Discovery, vol. 31, no. 5, pp. 1506–1543, 2017.
[27] C. Liu, J. Liu, and Z. Jiang, “A multiobjective evolutionary algorithm based on similarity for community detection from signed social networks.” IEEE Trans. Cybern., vol. 44, no. 12, pp. 2274–2287, Dec. 2014.
[28] Y. Li, J. Liu, and C. Liu, “A comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks,” Soft Comput., vol. 18, no. 2, pp. 329–348, 2014.
[29] R. Dunbar. Robin Dunbar on Dunbar Numbers. Accessed: Aug. 2017. [Online]. Available: http://www.sociablespace.com/2013/11/robin-dunbar-on-dunbar-numbers/
[30] Z. Ding, X. Zhang, D. Sun, and B. Luo, “Overlapping community detection based on network decomposition,” Sci. Rep., vol. 6, Apr. 2016, Art. no. 24115.
[31] A. Lázár, D. Ábel, and T. Vicsek, “Modularity measure of networks with overlapping communities,” EPL (Europhys. Lett.), vol. 90, no. 1, p. 18001, 2010.
[32] H. Shen, X. Cheng, K. Cai, and M. Hu, “Detect overlapping and hierarchical community structure in networks,” Phys. A, Stat. Mech. Appl., vol. 388, no. 8, pp. 1706–1712, 2009.
[33] S. Fortunato and M. Barthélémy, “Resolution limit in community detection,” Proc. Nat. Acad. Sci. USA, vol. 104, no. 1, pp. 36–41, 2007.

Weili Wu (M’00) received the M.S. and Ph.D. degrees in computer science from the University of Minnesota, Minneapolis, MN, USA, in 1998 and 2002, respectively. She is currently a Professor of computer science and the Director of the Data Communication and Data Management Laboratory with the University of Texas at Dallas, Richardson, TX, USA. She has authored or co-authored more than 258 research papers in various prestigious journals and conferences such as the IEEE TRANSACTION ON KNOWLEDGE DISCOVERY AND DATA ENGINEERING, the IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, the IEEE TRANSACTIONS ON MOBILE COMPUTING, the IEEE TRANSACTIONS ON MULTIMEDIA, ACM Transactions on Sensor Networks, the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE/ACM Transactions on Networking, Journal of Global Optimization, Journal of Optical Communications and Networking, Optimization Letters, IEEE COMMUNICATIONS LETTERS, Journal of Parallel and Distributed Computing, Journal of Computational Biology, Discrete Mathematics, Social Network Analysis and Mining, Discrete Applied Mathematics, IEEE INFOCOM (The Conference on Computer Communications), ACM SIGKDD (International Conference on Knowledge Discovery & Data Mining), International Conference on Distributed Computing Systems, International Conference on Database and Expert Systems Applications, SIAM Conference on Data Mining, UCGIS Summer Assembly, and International Conference on Computer Science and Informatics. Her current research interests include database systems, especially with applications in big data, social networks, geographic information systems and bioinformatics, distributed database in internet system, and wireless database systems with connection to wireless communication.

Dr. Wu is an Associate Editor of International Journal of Bioinformatics Research and Applications, and Computational Social Networks. She organized many conferences and edited several books/Special Issues of Journals.