Hidden Community Detection in Social Networks

Kun He∗
Huazhong University of Science and Technology
Wuhan, China
brooklet60@hust.edu.cn

Sucheta Soundarajan
Syracuse University
Syracuse, NY, USA
susounda@syr.edu

Yingru Li†
Huazhong University of Science and Technology
Wuhan, China
szrlee@hust.edu.cn

John E. Hopcroft
Cornell University
Ithaca, NY, USA
jeh@cs.cornell.edu

ABSTRACT
We introduce a new paradigm that is important for community
detection in the realm of network analysis. Networks contain a
set of strong, dominant communities, which interfere with the
detection of weak, natural community structure. When most of
the members of the weak communities also belong to stronger
communities, they are extremely hard to be uncovered. We call
the weak communities the hidden community structure.

We present a novel approach called HICODE (HIdden COmmu-
nity DEtection) that identifies the hidden community structure
as well as the dominant community structure. By weakening the
strength of the dominant structure, one can uncover the hidden
structure beneath. Likewise, by reducing the strength of the hidden
structure, one can more accurately identify the dominant structure.
In this way, HICODE tackles both tasks simultaneously.

Extensive experiments on real-world networks demonstrate that
HICODE outperforms several state-of-the-art community detection
methods in uncovering both the dominant and the hidden structure.
For example in the Facebook university social networks, we find
multiple non-redundant sets of communities that are strongly asso-
ciated with residential hall, year of registration or career position
of the faculties or students, while the state-of-the-art algorithms
mainly locate the dominant ground truth category. Due to the
difficulty of labeling all ground truth communities in real-world
datasets, HICODE provides a promising approach to pinpoint the
existing latent communities and uncover communities for which
there is no ground truth. Finding this unknown structure is an
extremely important community detection problem.

CCS CONCEPTS
-Information systems →Social networks; -Human-centered
computing →Social network analysis;

1 INTRODUCTION
Over the past decades, community detection has emerged as an
essential task in the realm of network analysis, and provides in-
sight into the underlying structure and potential functions of the
networks [14, 23]. Early work focused primarily on identifying
disjoint communities that partition the set of nodes within a net-
work [7, 29, 30]. More recently, researchers have observed the mul-
plicity of interwoven memberships of communities and have deve-
loped algorithms for finding overlapping communities [3, 10, 22, 37].
Some partitioning techniques are also extended to tackle the over-
lapping case [15, 17, 41]. Within these two categories, one can
further build a hierarchical dendrogram based on the granularity
of the detected communities.

Although much progress has been made, there is a type of com-

†Corresponding author.

Permission to make digital or hard copies of part or all of this work for personal or
classroom use is granted without fee provided that copies are not made or distributed
for profit or commercial advantage and that copies bear this notice and the full citation
on the first page. Copyrights for third-party components of this work must be honored.
For all other uses, contact the owner/author(s).
Submitted to a conference, Canada
© 2017 Copyright held by the owner/author(s). 978-x-xxxx-xxxx-x/YY/MM...$15.00
DOI 10.1145/nmnnnn.nmnnnn

© 2017 Copyright held by the owner/author(s). 978-x-xxxx-xxxx-x/YY/MM...$15.00
DOI 10.1145/nmnnnn.nmnnnn
Figure 1: The illustration of the dominant communities and the hidden communities in a social network. (a) The three cliques correspond to communities of students working closely as teams in projects. (b) The three groups of different colors correspond to sports communities with sparser connections.

As illustrated in Figure 1 for a small network, the hidden community structure is sparser and as it is tangled with the structure of the dominant communities, it is also harder for detection.

For applications in a large variety of scientific disciplines, the weak, hidden structure is of great interest and deserves to be explored. For example, in Protein-Protein Interaction (PPI) networks [28], biologists wish to identify gene groups serving similar functions. However, the current annotation is far from complete [12], and the dominant, clearest groupings are more likely to be annotated. In such a scenario, a way to help the biologists identify the hidden, less obvious groups is of great value. As another example, consider a government that wishes to identify communities of a criminal or terrorist organization in a social network. This community structure may be expressed within the network, but is likely to be much weaker than communities corresponding to family or location. Identifying this hidden structure in the presence of the stronger community structure is even more important but faces a major challenge.

This paper aims to provide insight into the hidden structure. We will address the following questions:

- Do real-world networks contain hidden community structure?
- How can we formally characterize “hidden” structure?
- How can one find hidden structure in the presence of stronger, dominant structure?

To address the above issues, we define the hiddenness value of a community as the portion of nodes in stronger communities, and present a meta-approach called Hidden Community Detection (HICODE) to identify the dominant structure as well as the hidden structure in networks. HICODE begins by first applying an existing algorithm as a base algorithm to a network, and then weakening the structure of the detected communities in the network. In this way, the weaker, hidden community structure becomes visible. This step is repeated iteratively until no further significant structure is detected. Next, HICODE weakens the structure of the hidden communities, and thus obtains a more accurate version of the dominant community structure.

Through experiments on real social networks, we demonstrate the existence of multiple sets of non-redundant communities. The weaker set of communities, though possessing significant community structure, is rarely detected by state-of-the-art community detection algorithms. On real-world networks containing a hidden ground-truth community structure, HICODE uncovers this structure much more precisely than the baseline algorithms.

Hidden community structure can be regarded as a special type of overlapping communities. However, existing overlapping community detection methods mainly focus on communities in which a considerable portion of the members are not “hidden”, that is, they could also belong to other weaker communities but this community is clearly the strongest for these members. We believe the insights we obtained on hidden community structure will provide valuable guidance for future investigations. The main contributions of this paper include:

- **Conception on Hidden Community.** We introduce the concept of hidden community structure that exists widely in social networks, and we formally define the hiddenness value of a community.

- **Methods on Hidden Community Detection.** We present HICODE for identifying both the dominant and the hidden structure. We implement HICODE with several community detection algorithms as the base algorithm, and provide several structure weakening methods: RemoveEdge, ReduceEdge and ReduceWeight.

- **Validation on real world datasets.** Through experiments on a variety of real-world networks, we demonstrate that the higher the hiddenness value a community is, the harder for an algorithm to locate such community; HICODE outperforms several state-of-the-art community detection methods on uncovering the hidden communities.

- **Scalability.** As a meta-approach, HICODE is scalable by applying any conventional community detection algorithm as the base algorithm. One can use HICODE as a tool to find overlapping communities by using any disjoint community algorithm as the base. In addition, by applying overlapping community detection algorithms as the base, HICODE can
improve the accuracy on the set of communities the base algorithm found, at the same time it can uncover more weak communities hidden underneath the dominant set.

2 PRELIMINARIES

Let graph $G = (V, E)$ represent a network with $n$ nodes and $e$ edges. Let $A$ be the adjacency matrix of $G$, the $ij$-entry $A_{ij} \in \{0, 1\}$ indicates whether there is an edge connecting nodes $i, j$. Let $C = \{C_1, C_2, …, C_K\}$ be all the overlapping communities, where $C_k = (V_k, E_k)$ is a comparatively dense subgraph of $G$. The cardinality of a community is defined as the number of nodes in this community, i.e. $|C_k| = |V_k|$. We first introduce some necessary metrics, then provide formal definitions on the hidden structure.

2.1 Metrics

Modularity. To measure the strength of a set of communities that partition the network, we adopt the popular modularity metric. Modularity is defined by Newman as the ratio of the number of intra-community edges to the expected number of edges in the same set of communities if the edges had been distributed randomly while preserving degree distribution [25].

The modularity score $Q$ of a partition is calculated by:

$$Q = \frac{1}{\sum_{k=1}^{K} |C_k|} \sum_{k=1}^{K} \left[ \frac{\sum_{i < j \in C_k} \epsilon_{kk} \epsilon_{ij} - \left( \frac{\sum_{i < j \in C_k} \epsilon_{kj} \epsilon_{ij}}{2\epsilon_{kk}} \right)^2}{\epsilon_{kk}} \right],$$

where $K$ is the number of communities in the partitioning, $e$ is the number of edges in the graph, $\epsilon_{kk}$ is the number of edges within community $C_k$, and $d_k$ is the total degree of the nodes in community $C_k$. $Q$ lies in the range $[-0.5, 1]$, with larger values indicating a stronger community set.

Definition 2.1. Modularity of a Community. We call $Q_k$ the sum modularity contribution of $C_k$, and the modularity of a single community is defined as the sum modularity contribution of that community divided by the number of nodes in the community, i.e. $Q_k/|C_k|$.1

Zhang et al. extend Newman’s definition to a set of overlapping communities by considering the belonging coefficient $w_{ik}$ for node $i$ to community $C_k$ [41]. Briefly, $w_{ik} = 1/m$ if community $C_k$ is one of the $m$ communities containing node $i$. Then in Eq. (1), $\epsilon_{kk}$ is weighted by

$$\epsilon_{kk} = \frac{1}{2} \sum_{i \neq j \in C_k} w_{ik} + w_{jk} - A_{ij},$$

where $A_{ij}$ is the $ij$-entry of the adjacency matrix. We know $d_k = 2\epsilon_{kk} + \epsilon_{k, out}$, then $\epsilon_{k, out}$ is weighted by

$$\epsilon_{k, out} = \frac{1}{2} \sum_{i \in C_k, j \notin C_k} w_{ik} + (1 - w_{jk}) - A_{ij}.$$

The extended definition degenerates exactly to Eq. (1) for disjoint communities.

Normalized Mutual Information (NMI). We use normalized mutual information (NMI) to capture the similarity of two partitions $X$ and $Y$ [11].

$$NMI(X, Y) = \frac{2I(X, Y)}{H(X) + H(Y)},$$

where $H(X)$ is the Shannon entropy of partition $X$, and $I(X, Y)$ is the mutual information that captures the similarity between two partitions $X$ and $Y$. For a set of communities that partition a network, $p(x) = |x|$, and $p(x, y) = |x \cap y|$. The NMI score lies in the range $[0, 1]$, where 1 represents a perfect matching and 0 indicates total independence. For overlapping communities, we use the extended NMI [1]. See [1] for details.

$$I(X, Y) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)}$$

$$H(X) = -\sum_{x \in X} p(x) \log p(x)$$

2.2 Definitions

Assume we have some metric function $\mathcal{F}(G, C_k) \rightarrow \mathbb{R}$ (e.g., modularity [25] or conductance [32]) that assigns a quality score to a community. For simplicity, let $Y_k$ denote the quality of $C_k$ in $G$. Here, we assume that higher scores indicate stronger communities, but the definition below can be trivially modified for the case when lower scores indicate higher community quality.

Definition 2.2. Hiddenness Value of a Community. The hiddenness value $H(C_k)$ of community $C_k$ is the fraction of nodes of $C_k$ belonging to various communities with a higher $\mathcal{F}$ score.

Let $S_k$ be the set of all strong communities for community $C_k$.

$S_k = \{C|\mathcal{F}_i > \mathcal{F}_k, C \in C\}$

The hiddenness value of $C_k$ can be calculated as:

$$H(C_k) = \frac{1}{|C_k|} \sum_{C \in S_k} |C \cap C_k|$$

$H(C_k) \in [0, 1]$. Intuitively, the higher a community’s hiddenness value is, the more difficult for the community to be uncovered.

The goal of the hidden community detection problem is to locate overlapping communities in the network such that communities having high hiddenness values can be found. Note that there is no single threshold for a ‘high’ hiddenness value, these values depend on the network under study, the metric being used, and the set of communities. For two communities $C_i, C_j \in G$, if $H(C_i) > H(C_j)$, then $C_i$ is comparatively hidden as compared with $C_j$, and $C_j$ is comparatively dominant as compared with $C_i$.

For convenience, we separate the communities into layers.

Definition 2.3. Layer. A layer is a set of communities that partitions or covers the nodes of the network.2

Intuitively, in a social network, a partitioning layer may correspond to a grouping of individuals by their locations, and colleague circles could form an overlapping layer where some colleagues are in interdisciplinary areas.

Definition 2.4. Hiddenness Value of a Layer. The hiddenness value $H(L_i)$ of a layer $L_i$ is the weighted average hiddenness values of the communities in this layer.

$$H(L_i) = \frac{\sum_{C_k \in L_i} |C_k| \cdot H(C_k)}{\sum_{C_k \in L_i} |C_k|}.$$

1For simplicity, we discuss unweighted graph, but all discussions in this paper are easily extended to weighted graph by letting $A_{ij} \in [0, 1]$ to indicate the weight of each edge. In the weighted graph, the number of edges then corresponds to the sum of the edge weights.

2Here we allow trivial communities of size less than 3.
A layer is called hidden if it has a comparatively high hiddenness value. Figure 1 illustrates a small social network with two layers of communities, the dominant layer corresponds to project teams and the hidden layer corresponds to sports groups.

2.3 Case Study
To illustrate the concept of hidden community structure, we choose modularity as the community metric, and show an example on the Caltech Facebook network dataset (described in detail in Section 4.3). For simplicity, here we only consider two annotations for each individual: the residence hall ("Dorm") and year of matriculation ("Year"). A ground truth community contains all individuals living in the same dormitory, or are in the same year of matriculation. "Dorm" communities partition the nodes and form a layer. Similarly, "Year" communities form another layer.

On average, 79% of the nodes in each "Year" community belong to a stronger "Dorm" community, i.e., the "Year" layer has a hiddenness value of 0.79. In contrast, the "Dorm" layer only has a hiddenness value of 0.08. The "Year" layer have substantially higher hiddenness values than the "Dorm" layer. Note that "Year" is still the hidden layer as compared with "Dorm" when we use the conductance metric. We thus conjecture that standard community detection algorithms have difficulty detecting the "Year" layer. As we will see in the experiments, the state-of-the-art community detection algorithms that we consider can effectively find the "Dorm" layer but rarely detect the "Year" layer, while HICODE can effectively uncover both layers.

3 HIDDEN COMMUNITY DETECTION

3.1 Algorithm Overview
In this section, we propose a meta-approach called HICODE to find layers of community structure in a network. HICODE uses an existing community detection algorithm as the ‘base’ method, and iteratively weakens the structure of detected layers to reveal hidden structure. HICODE then applies a refinement procedure to increase the quality of the layers.

HICODE contains two stages: Identification and Refinement.

Stage 1. Identification:
The Identification stage determines the initial layers of communities as follows:

1. Identify a layer of communities via the base method;
2. Weaken the structure of the detected layer;
3. Repeat until the appropriate number of layers are found.

In this stage, we iteratively identify a set of initial layers by weakening the structure of the previous, stronger layers.

Section 3.2 presents methods for weakening a detected community structure in order to reveal the hidden structure beneath. A crucial aspect of the Identification stage is to automatically determine the number of layers \( n_L \) in a network. This is accomplished by increasing \( n_L \) until a stopping condition is met. We describe this process in Section 3.3.

Stage 2. Refinement:

It is reasonable that stronger community structure can obscure weaker community structure, but critically, we observe that weaker structure can also hinder the accurate or complete detection of the stronger structures. After the Identification stage, one has only a rough approximation of the various community layers, and the purpose of the Refinement stage is to further improve the quality of these detected layers.

Refinement is an iterative process. In each iteration, we consider each layer, and improve the current layer as follows:

1. Weaken the structures of all other layers from the original network to obtain a reduced network;
2. Apply the base algorithm to the resulting network.

In contrast to the Identification stage, where only the layers found so far (i.e., the stronger layers) are reduced, during the Refinement stage, we weaken the effects of both the stronger and weaker layers. This is necessary because the weaker layers can impair detection of the layer currently under consideration, even though they have a smaller impact on the network structure than the stronger layers. Through this process, a more accurate version of the current layer is produced.

We use NMI [1, 11] to capture the similarity of two partitions or two sets of overlapping communities. For the synthetic data that we consider, the Refinement stage quickly converges within 30 iterations, meaning that the NMI of the corresponding layers at adjacent iterations is almost 1. For real-world datasets, 100 iterations is generally enough.

Although the trend over these iterations is to see higher quality community layers (i.e., higher modularity partitions or coverings), there are fluctuations in this trend. In each Refinement iteration, we calculate the modularity of the detected layers, and the final output corresponds to the iteration with the highest average modularity score.

3.2 Reducing Methods
We present the following methods to reduce a single layer of community structure: RemoveEdge, ReduceEdge, and ReduceWeight. RemoveEdge works reasonably well when there are only a small number of layers, and communities in different layers have comparatively small overlaps. ReduceEdge generally performs better, but it typically does not perform as well as ReduceWeight; however, one major advantage that ReduceEdge has over ReduceWeight is that it does not require a base algorithm that supports weighted networks. As we will see in experiments, ReduceWeight is the best-performing method of the three.

For all cases, if one has a layer of overlapping communities, then in order to avoid duplicate weakening on the overlapping portions, we sort the communities according to their sizes, weaken larger communities first and do not weaken the overlapping portion again for subsequent communities in the same layer.

3.2.1 RemoveEdge. RemoveEdge weakens a detected community by removing all intra-community edges. This method is inspired by algorithms that find communities by removing edges, such as the classic Girvan-Newman algorithm, which, removes edges with high betweeness [14]. RemoveEdge works reasonably well when there are few layers and communities in different layers have comparatively small overlaps.
3.2.2 ReduceEdge. ReduceEdge approximates each layer as a single stochastic blockmodel, where other edges are regarded as background noise. This method randomly removes some edges within each community block so that the edge probability in the block matches the background edge probability of this block.

For each community $C_k$ in a single layer that we want to reduce, we calculate the observed edge probability $p_k$ in $C_k$, and the background block probability $q_k$, as illustrated in the reordered adjacency matrix in Figure 2. Let $n_k$ and $e_{k,k}$ be the number of nodes and edges inside $C_k$, and $d_k$ the sum of degrees of nodes in $C_k$. $p_k$ equals the actual number of edges in $C_k$ divided by the maximum possible number of edges:

$$p_k = \frac{e_{k,k}}{0.5n_k(n_k - 1)}, \quad (10)$$

and $q_k$ is the average outgoing edge density of block $C_k$:

$$q_k = \frac{d_k - 2e_{k,k}}{n_k(n_k - n_k)}.$$

ReduceEdge removes each edge within community $C_k$ with probability $1 - q_k'$. In this way, edges are randomly removed from $C_k$ such that the edge probability within $C_k$ matches $q_k$.\footnote{Note that ReduceEdge is also suitable for weighted graph where $e_{k,k}$ corresponds to the sum of edge weights inside $C_k$, and $d_k$ corresponds to the sum of degrees of nodes in $C_k$, weighted by the edge weight.}

3.2.3 ReduceWeight. This method reduces the weight of each edge within community $C_k$ by a factor of $q_k'$, defined in Eq. (12). Like with ReduceEdge, we wish to set the weight of each edge within $C_k$ equal to the average weighted background probability $q_k$.

Note that to use ReduceWeight, one’s base algorithm must support weighted networks. However, the original network itself need not be weighted, as one can simply set the original weight of every edge to 1. We let ReduceWeight degenerate to ReduceEdge if the base algorithm does not support weighted graphs. Unlike ReduceEdge, ReduceWeight is deterministic.

3.3 Selecting the Number of Layers

A major challenge for HICODE is determining $n_L$, the appropriate number of community layers. On synthetic networks (details of the data model are described Section 4.2) where all communities are labeled, we observe that if $n_L$ is chosen correctly, then during the refinement stage, the average modularity of the detected layers increases. If $n_L$ is either over or under-estimated, then this trend declines. Our rule for determining the number of layers $n_L$ is motivated by this observation: if one selects the appropriate number of layers, the output will generally be of a higher quality.

We begin by setting the number of layers $n_L = 2$, and increase the number of layers until a stopping condition is met. For each candidate number of layers $n_L$, HICODE first calculates the modularity of the weakest layer obtained at the Identification stage. If the value is very low, then there are no more significant layers, so we set $n_L = n_L - 1$ and return; otherwise, let $Q_t$ be the average modularity of all the detected layers at step $t$:

1. Calculate $Q_0$ for $t = 0$, i.e. after identification, before any refinement is conducted;
2. Perform T = 10 tentative iterations of refinement, and calculate $Q_t$ for each $t \in \{1, ..., T\};$
3. Calculate the average improvement ratio of modularity per iteration\footnote{Instead of only considering how much improvement we get at step $T$, we use the average modularity of the $T$ steps to balance the fluctuation on real-world networks.}: $R_T = \frac{\sum_{t=1}^{T} Q_t}{TQ_0}$.

$R_T$ represents how much refinement improves the detected layers. If $n_L$ is too high, then when we remove the structure of the extra layers, we will be removing structure that actually belongs to some earlier layer. This will result in a lower quality partition, such that the refinement stage actually lowers the quality of the detected layers. Thus, we choose $n_L$ corresponding to the peak $R_T$.

4 EXPERIMENTAL SETUP

4.1 Evaluation Metric

Besides NMI, we also define a Jaccard score-based metric to evaluate how well the detected layers resemble the ground truth communities. Given a set of detected communities $D$ and a set of ground truth communities $G$, the Jaccard similarity-based precision, recall, and $F_1$ score are defined as follows:

Each detected community $D_i$ has its individual Jaccard Precision

$$P(D_i) = \max_{G_j \in G} \frac{|G_j \cap D_i|}{|G_j| \cup |D_i|}.$$  \quad (1)

The Jaccard Precision $P$ of the set $D$ is defined as the weighted average of $P(D_i)$ over all detected communities, weighted by the size of the communities.

Each ground truth community $G_j$ has its individual Jaccard Recall

$$R(G_j) = \max_{D_i \in D} \frac{|G_j \cap D_i|}{|G_j| \cup |D_i|}.$$  \quad (2)

The Jaccard Recall $R$ of the set $G$ is defined as the weighted average of $R(G_j)$ over all ground truth communities, weighted by the size of these communities.

We use weighted $P$ and $R$ to give a higher priority to bigger communities. The $Jaccard F_1$ Score is defined as the harmonic mean of $P$ and $R$, i.e.

$$F_1 = 2PR/(P + R).$$

4.2 Synthetic Stochastic Blockmodel

We define a synthetic stochastic blockmodel containing multiple layers of planted communities. Each layer in a network corresponds to a single stochastic blockmodel that partitions the nodes into roughly equally-sized sets. In each layer, we first create the
appropriate number of community IDs and randomly assign each node to a community, then we produce a $G(n, p)$ Erdos-Renyi random graph over each block. We select suitable $p$-values for the different layers so that they are of different densities, but roughly equal strengths as measured by modularity. By randomly assigning nodes to communities, we ensure that the communities in different layers are independent (i.e., a node’s membership in different layers is unrelated).

We then generate a small synthetic network SynL2 that will be used to illustrate the details of our algorithm. There are two layers of communities, containing 5 communities of roughly size 40, and 4 communities of roughly size 50. The $p$ values that govern the number of intra-community links for these two layers are 0.12 and 0.10. Their modularity scores are 0.40 and 0.39, with hiddenness values of 0.43 and 0.56. We observe a very low $NMI$ of 0.05 between the two layers, and the $F_1$ score is also low at 0.02.

### 4.3 Real-world Datasets

We do thorough testing on two groups of real-world networks, as summarized in Table 1.

1. **Facebook networks**: These networks are portions of the Facebook social network for different universities in the United States [33], which comes from a single-day snapshot in September 2005 when one needed a ‘.edu’ email address to become a member of Facebook. For each university network, the data have categorical attributes encompassing the gender, major, year of matriculation, high school, dormitory, status (faculty, student, staff, etc.) and residence (house, dormitory, fraternity, etc.) of the users. We choose eight representative university networks which express communities of nodes.

2. **SNAP networks**: We choose three networks with ground-truth communities collected by SNAP$^3$ [38]: Youtube, Amazon and DBLP. Youtube indicates friendships among users and the ground truth communities are user-defined groups. Amazon is a product co-purchasing network where co-purchased products are connected by edges, and the ground truth corresponds to product categories. DBLP is a co-authorship network where nodes represent authors, edges represent the co-authorship, and the ground truth communities correspond to conferences. For each network, we extract a subnetwork (*For convenience, we still use the same name*) containing all nodes in the top 5000 ground-truth. For further analysis, we partition the ground-truth communities into several community sets ($CommSet$) based on their hiddenness values.

More statistics for their ground-truth communities are as shown in Table 2 (left column): the modularity and average hiddenness value for each ground truth community layer, as well as the maximum $NMI$ ($NMI_{max}$), maximum and average $F_1$ scores ($F_{max}$ and $F_{avg}$) for pairwise layers.

For Facebook networks, each layer gives rise to a set of communities grouped by a common attribute (i.e., nodes with a common annotation are in the same community), and we call each such set of communities an annotated layer (e.g., the ‘Dorm’ annotation gives rise to one annotated layer). These annotated layers cover all nodes (coverage = 100%) in the corresponding networks.

### 5 EXPERIMENTAL RESULTS

We first evaluate different reducing methods of HICODE and illustrate the necessity of the Refinement stage. Then we show the statistics of the multiple layers detected by HICODE and compare the detection accuracy with state-of-the-art baselines on real-world networks.

#### 5.1 Analysis on HICODE

**5.1.1 Comparison of the Reduction Methods**: We first evaluate different reducing methods on each of the versions of HICODE (corresponding to different base algorithms). Figure 3 shows the Recall for all layers of communities detected by HC:M/0, HC:M, HC:OS and HC:LC on several examples of the Facebook networks: Caltech, Smith, Rice and Vassar.

---

1. http://snap.stanford.edu
2. Code and synthetic data: https://github.com/KunHe2015/HCode/
which forms a positive feedback cycle. The refinement process with each community in layer 1. Figures 4 and 5 illustrate several right column shows the statistics of the layers found by multiple layers of high modularity community structure. Table 2 For real-world networks from various domains, 5.2 Evaluation on Real Networks trend during the iteration when we detect 2, 3, 4 or 5 community Refinement stage. As an example, Figure 6 shows the increasing in further iterations. times in other community layers, we get a more accurate current layer, then during the Refinement stage, by iteratively weakening the less stronger layer 2 after the initial detected layer 1 is weakened. Roughly identifies the stronger layer 1, and only roughly identifies the less stronger layer 2 after 7 In general, ReduceWeight reaches the highest detection accuracy, followed by ReduceEdge and RemoveEdge. ReduceWeight and ReduceEdge provide a more accurate estimation on the background density, while RemoveEdge removes all intra-community edges and impairs more structure of other layers, especially when communities in different layers overlap considerably. ReduceWeight deterministically reduces the weight of intra-community edges and outperforms ReduceEdge, which randomly removes intra-community edges. To save space, we will present results using ReduceWeight. For the different versions of HICODE, HC:MO.Sc and HC:IM perform consistently better than HC:OS and HC:LC. We will use HC:MO.Sc to show the property of HICODE, and then compare our results on the four HICODE implementations with other algorithms.

5.1.2 Necessity of the Refinement Stage. To show the necessity of the Refinement stage, which gradually separates the community structures mixed together and strengthens the structure of each layer, we run HC:MO.Sc on the small network SynL2 and illustrate its results.7

There are two planted layers on SynL2, with hiddenness values of 0.45 and 0.56. Communities in layer 2 is comparatively more hidden with respect to communities in layer 1. On average, 20% of the nodes in a community of layer 1 overlap with each community in layer 2, and 25% of the nodes in a community of layer 2 overlap with each community in layer 1. Figures 4 and 5 illustrate several snapshots of the detected two layers during the execution (shown by the adjacency matrix but the node IDs are reordered for the two layers respectively). In the Identification stage (t = 0), HICODE only roughly identifies the stronger layer 1, and only roughly identifies the less stronger layer 2 after the initial detected layer 1 is weakened. Then during the Refinement stage, by iteratively weakening the other community layers, we get a more accurate current layer, which forms a positive feedback cycle. The refinement process converges within 20 steps and the two detected layers remain stable in further iterations.

In addition, we also observe that the average modularity scores of the detected layers on the original network are improved by the Refinement stage. As an example, Figure 6 shows the increasing trend during the iteration when we detect 2, 3, 4 or 5 community layers on four real-world networks.

5.2 Evaluation on Real Networks
For real-world networks from various domains, HICODE uncovers multiple layers of high modularity community structure. Table 2 (right column) shows the statistics of the layers found by HC:MO.Sc

Figure 3: Stacked Recall of all Layers for the reducing methods (left to right: RemoveEdge, ReduceEdge, ReduceWeight).

Figure 4: Refinement of layer 1 on SynL2.

Figure 5: Refinement of layer 2 on SynL2.

in each network. When comparing these layers to one another, their maximum pairwise NMI, maximum and average pairwise F1 scores are low, indicating that these layers of communities are distinct. One exception is Amazon, which seems only to contain one layer of very strong community structure.

When comparing the ground truth layers on the Facebook networks, we see the layers found by HICODE are strongly associated with the community categories. Tables 3 show the F1 and NMI scores of HICODE and the baseline algorithms evaluated against different annotation categories on several examples: Caltech, Smith, Rice and Vassar. We see that:

(1) All baseline algorithms primarily locate the dominant structure, and rarely detect the hidden, weaker structures. Interestingly, different to other algorithms, LC regards Status and Year as the dominant on Vassar and Smith respectively. This occurs as LC may have a different notion of community structure.

(2) All HICODE implementations return community layers that are strongly associated with each ground truth category. HICODE not only uncovers the hidden layers that the baseline algorithms rarely detect, but also improves the detection accuracy on the dominant layer.

For each CommSet of the SNAP networks, we calculate the Recall of AllLayers (the union of all layers found by HICODE) and the Recall of the communities detected by each of the baselines. In Table 4, we see that the comparatively hidden CommSetA is consistently harder to be located than the dominant CommSetA. When compared with the baselines, HICODE not only has higher detection accuracy on the comparatively hidden CommSetA, but also boost the detection accuracy on the comparatively dominant communities in CommSetA. Again, on Amazon, all algorithms detect both

7The other base algorithms produce similar results.
CommSet_A and CommSet_B fairly well due to the highly similarity of the two sets of ground truth communities.

6 RELATED WORK

In the past decades, a plethora of community detection algorithms have been presented for uncovering such latent modular structure based on different metrics (e.g., modularity [18, 24, 25], conductivity [6, 32, 38], etc.) and techniques (e.g., random walk [5, 30], heat kernel [20], spectral clustering [35], seed set expansion [21, 27, 34], distance dynamics [31], mixed membership stochastic blockmodels [2, 4, 19], etc.). Still, there are many new technologies and directions emerging, such as the local community detection [17, 20, 21].

To the best of our knowledge, this is the first work to formally propose and address the hidden community detection problem. Due to space limit, here we only highlight several related works in the area of clustering and community detection. For comprehensive reviews to various community detection algorithms and techniques, please refer to survey papers such as [9, 36].

Outside the realm of community detection, researchers have studied the problem of clustering data into multiple alternative groupings. By adopting orthogonal clustering and clustering in orthogonal subspaces, Cui et al. [39] cluster the data points in different views, where data points of one cluster can belong to different clusters in other views. By augmenting a spectral clustering objective function to incorporate dimensionality reduction and multiple views, and to penalize for redundancy between the views, Niu et al. proposed approaches to learn non-redundant subspaces that provide multiple views simultaneously [13] and iteratively [26]. In contrast to our work, note that their work is not about finding hidden structure, but rather high-quality orthogonal clusterings (possibly in a projected space), which may or may not be hidden by stronger clusterings.

In the realm of community detection, there are several pioneer and embryonic works. Yang and Leskovec [37] found that community overlaps are more densely connected than the non-overlapping parts, which coincide with our assumption that the overlapping part are the superposition of the layers. And they propose a Community-Affiliation Graph model to maximize the affiliation likelihood. Then, there are three pieces of related work that are proposed independently and almost simultaneously [8, 16, 40].

Chen et al. [8] remove nodes or edges based on the local Fiedler vector centrality (LFVC) which is associated with the sensitivity of algebraic connectivity to node or edge removals. Most importantly, they define a concept of deep community as a connected component that can only be seen after removal of all nodes or edges from the rest of the network. They prove that their method works on small...
Hidden Community Detection in Social Networks

Submitted to a conference, 2017, Canada

In this paper, we propose a new paradigm of hidden community structure for network analysis and community detection. We present a formal definition on ‘hidden structure’, and propose a systematic method called HICODE to uncover these hidden communities. Through extensive experiments, we demonstrate that hidden communities exist in real-world networks of various domains, and the proposed method significantly outperforms several comparative methods, including overlapping community detection algorithms, in accurately locating these hidden structures. Our work sheds light on the organization of complex networks and provides new directions for research on community detection.

In the future, we plan to explore the hidden communities on directed or weighted networks, apply other community detection algorithm as the base, as well as design new reduction method for weakening the detected communities. It is also possible that our new ideas can be applied to other network mining tasks such as link-prediction or maximum influence propagation.

ACKNOWLEDGMENTS

The work is supported by US Army Research Office (No. W911NF-14-1-0477), National Science Foundation of China (No. 61472147), Microsoft Research Asia Collaborative Research (No. 97354136), and National Science Foundation of Hubei Province (No. 2015CFFB566).
Submitted to a conference, 2017, Canada

REFERENCES

[1] Derek Greene Aaron F. McDaid and Neil Hurley. 2013. Normalized Mutual Information to evaluate overlapping community finding algorithm. arXiv:1110.2515v2 (2013).

[2] Emmanuel Abbe and Colin Sandon. 2015. Recovering communities in the general stochastic block model without knowing the parameters. In NIPS 28: 676–684.

[3] Yong-Yeol Ahn, James P. Bagrow, and Sune Lehmann. 2010. Link communities reveal multiscale complexity in networks. Nature 466, 7307 (2010), 761–764.

[4] Edoardo A. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. 2008. Mixed membership stochastic blockmodels. Journal of Machine Learning Research 9 (2008), 1981–2014.

[5] Reid Andersen, Fan Chung, and Kevin Lang. 2006. Local graph partitioning using PageRank vectors. In FOCS. 475–486.

[6] R. Andersen and K. J. Lang. 2006. Communities from seed sets. In WWW. ACM, 223–232.

[7] Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment 10 (2008), P10008.

[8] Pin-Yu Chen and Alfred O. Hero III. 2014. Deep Community Detection. CoRR abs/1407.6071 (2014).

[9] Michele Coscia, Fosca Giannotti, and Dino Pedreschi. 2011. A classification for community discovery methods in complex networks. Statistical Analysis and Data Mining 4, 5 (2011), 512–546.

[10] Michelle Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi. 2012. DEMON: a local-first discovery method for overlapping communities. In KDD. ACM, 615–623.

[11] Leon Danon, Albert Díaz-Guilera, Jordi Duch, and Alex Arenas. 2005. Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment 9 (2005), P09008.

[12] Jishnu Das and Haiyuan Yu. 2012. HINT: High-quality protein interactomes reveal multiscale complexity in networks. Nature 466, 7307 (2010), 761–764.

[13] Martin Rosvall and Carl T. Bergstrom. 2008. Maps of random walks on complex networks reveal community structure. Proceedings of the National Academy of Sciences 105, 4 (2008), 1118–1123.

[14] Junming Shao, Zhihao Han, Qunli Yang, and Tao Zhou. 2015. Community Detection based on Distance Dynamics. In KDD. 1075–1084.

[15] Jennifer G. Dy Ying Cui, Xiaoli Z. Fern. 2007. Non-Redundant Multi-View Clustering Via Orthogonalization. In KDD. 133–142.

[16] Jean-Gabriel Young, Antoine Allard, Laurent Hebert-Dufresne, and Louis J. Dube. 2015. A shadowing problem in the detection of overlapping communities: Lifting the resolution limit through a cascading procedure. PLoS ONE 10(10) (2015), e0140133.

[17] Shihua Zhang, Ruisheng Wang, and Xiangsun Zhang. 2007. Identification of overlapping community structure in complex networks using fuzzy c-means clustering. Physica A: Statistical Mechanics and its Applications 374, 1 (2007), 483–490.