A Community-Aware Framework for Social Influence Maximization

Abhishek K. Umrawal, Christopher J. Quinn, *Member, IEEE*, and Vaneet Aggarwal, *Senior Member, IEEE*

***Abstract*—We consider the problem of *Influence Maximization* (IM), the task of selecting k seed nodes in a social network such that the expected number of nodes influenced is maximized. We propose a community-aware divide-and-conquer framework that involves (i) learning the inherent community structure of the social network, (ii) generating candidate solutions by solving the influence maximization problem for each community, and (iii) selecting the final set of seed nodes using a novel progressive budgeting scheme.**

Our experiments on real-world social networks show that the proposed framework outperforms the standard methods in terms of run-time and the heuristic methods in terms of influence. We also study the effect of the community structure on the performance of the proposed framework. Our experiments show that the community structures with higher modularity lead the proposed framework to perform better in terms of run-time and influence.

***Index Terms*—Social networks, influence maximization, viral marketing, community detection, submodular maximization**

I. Introduction

A. Motivation

THE advent of social media has changed how traditional marketing strategies were used to be designed [1]. Companies are now preferring to allocate a significant proportion of their marketing budget to drive sales through large social media platforms. There are several ways in which social media can be leveraged for promotional marketing. For instance, advertising on the most visited social platforms, making social media pages for branding and spreading the word about the product, etc. A more sophisticated approach for promotional marketing would be to use the dynamics of the social network to identify the right individuals to be incentivized to get the maximum influence in the entire network.

A. K. Umrawal is with the School of Industrial Engineering, Purdue University, West Lafayette, IN, 47907, USA (email: aumrawal@purdue.edu), and Department of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, Baltimore, MD, 21250, USA. C. J. Quinn is with the Department of Computer Science, Iowa State University, Ames, IA, 50011, USA (email: cjquinn@iastate.edu). V. Aggarwal is with the School of Industrial Engineering, and Elmore Family School of Electrical and Computer Engineering, Purdue University, West Lafayette, IN, 47907 USA (email: vaneet@purdue.edu). He is also with Computer Science, KAUST, Thuwal, 23955, KSA.

This material is based upon work supported in part by the National Science Foundation under Grants No. 1742847, 2149588, and 2149617.

This paper has been accepted for publication in IEEE Transactions on Emerging Topics in Computational Intelligence (TETCI) in Dec 2022.

In the context of social media marketing, Domingos and Richardson posed the *Influence Maximization* (IM) problem [2]: "if we can try to convince a subset of individuals in a social network to adopt a new product or innovation, and the goal is to trigger a large cascade of further adoptions, which set of individuals should we target?" Formally, it is the task of selecting k seed nodes in a social network such that the expected number of influenced nodes in the network (under some influence propagation model), referred to as the *influence*, is maximized. Kempe et al. [3] showed that the problem of influence maximization is NP-Hard. This problem has been widely studied in the literature and several approaches for solving it have been proposed. Some approaches provide near-optimal solutions but are costly in terms of run time. On the other hand, some approaches are faster but heuristics, i.e. do not have approximation guarantees.

Motivated by the idea of addressing this trade-off between accuracy and run-time, we propose a community-aware divide-and-conquer framework to provide a time-efficient solution. The proposed framework outperforms the standard methods in terms of run-time and the heuristic methods in terms of influence.

B. Literature Review

Researchers have proposed different algorithms and heuristics for solving the Influence Maximization (IM) problem using several approaches. We now discuss several categories of the relevant approaches as follows. We refer to methods that presume knowledge of the network and estimate influence using Monte Carlo simulations of the diffusion process as *simulation-based methods*.

1) Simple heuristics: Degree centrality is perhaps the simplest way to quantify the influence of an individual in the network [3]. Observing the fact that many of the most central nodes may be clustered, targeting all of them is not at all necessary, Chen et al. [4] proposed the degree discount heuristic. These heuristics are simple and time-efficient. However, they do not have any provable guarantees.

2) Simulation-based methods: Under the *independent cascade* [5], [6] and *linear threshold* [7], [8] models of diffusion (discussed in Section II-B), Kempe et al. [3] showed that the problem of influence maximization is NP-Hard. They also proposed to use an efficient greedy algorithm [2] which due to a result by Nemhauser et al. [9] gives an $\left(1-\frac{1}{e}\right)$-approximation of the solution. The asymptotic run-time of this algorithm is O(nk). Asymptotically, this greedy algorithm is efficient but empirically the costly Monte Carlo simulations

cause an overhead. Leskovec et al. [10] proposed the CELF algorithm which improves upon the empirical run-time of the simple greedy algorithm by further exploiting the property of submodularity. Goyal et al. [11] proposed the CELF++ algorithm which further improved upon the empirical run-time of the CELF algorithm by even further exploiting the property of submodularity to avoid unnecessary re-computations of marginal gains incurred by CELF. Borgs et al. [12] proposed a greedy algorithm using reverse influence sampling (RIS) – an approach to efficiently estimate the influence of a seed set. CELF, CELF++, and [12] have the same worst-case run time O(nk) and approximation ratio $\left(1-\frac{1}{e}\right)$ as the one proposed by Kempe et al. [3]. Lotf et al. [13] proposed a genetic algorithm-based heuristic algorithm for dynamic (evolving over time) networks. This method involves Monte Carlo simulation and does not have any approximation guarantees. The framework proposed in this paper may also involve Monte Carlo simulations. But, the divide-and-conquer strategy allows us to significantly reduce the run-time.

3) Community-based methods: As the proposed method utilizes the inherent community structure of the network, we discuss other community-based methods of influence maximization as follows. Chen et al. [14] proposed two methods called CDH-KCut and CDH-SHRINK under heat diffusion model [15]. They further improved their methods and proposed another method called CIM [16]. Bozorgi et al. [17] proposed a method called INCIM which works only for the linear threshold diffusion model. Moreover, the method involves overlapping community detection contrary to our work where the communities are non-overlapping. Bozorgi et al. [18] have also developed a method for competitive influence maximization [19] under the competitive linear threshold model. Shang et al. [20] have proposed a method called CoFIM under the independent cascade diffusion model and weighted cascade edge-weight model. Contrary to these methods, our method does not depend on the choice of the diffusion model. Huang et al. [21] proposed a data-based method called CTIM which requires a potential action log and item-topic relevance.

4) Data-based methods: Provided some observational data involving real-world diffusion traces is available, the Monte Carlo simulations can be avoided by estimating the influence directly from the data. Goyal et al. [22], instead of using a propagation model, proposed a data-based-method to introduce a model called the credit distribution model, which directly leverages the propagation traces from real-world data and learns the flow of influence in the network. Pen et al. [23] and Deng et al. [24] have studied variants of the credit distribution model under time constraints and node features respectively. The proposed method does not involve any observational data.

5) Online methods: More recently, the focus has been on solving the problem of influence maximization in an online manner where the goal is to maximize the cumulative observed influence of the seed sets chosen at different times while receiving instantaneous feedback. Approaches differ based on semi-bandit feedback [25]–[29] and full-bandit feedback [30], [31]. The proposed method is not an online method.

C. Contribution

In Section I-B, we discussed that the CELF++ [11] algorithm is faster compared to the simple greedy algorithm [2], [3]. But the costly aspect of performing a large number of diffusions in the entire network is still there. Motivated by the idea of solving the influence maximization problem in a time-efficient manner, we propose a community-aware divide-and-conquer framework that involves (i) learning the inherent community structure of the social network, (ii) generating candidate solutions by solving the influence maximization problem for each community, and (iii) selecting the final set of individuals to be incentivized from the candidate solutions using a novel progressive budgeting scheme. Our method may also use the Monte Carlo simulations but we are restricting them within each community as compared to the entire network which brings savings in terms of run-time as compared to the CELF++ algorithm.

Compared to the other community-based methods, the proposed framework is novel in the following ways. It is not limited to a specific diffusion and/or an edge-weight model. In Step 1, the set of candidate solutions is generated by all combinations of solutions from each community. In Step 2, the final seed selection is performed by solving an integer linear program (ILP) over candidate solutions subject to a budget constraint. We propose an efficient progressive budgeting scheme to efficiently solve the ILP in Step 3. We provide the proof of correctness of this scheme which leverages submodularity (defined in Section II) of the influence.

We provide experiments on real-world social networks, showing that the proposed framework outperforms simulation-based methods in terms of run-time and heuristic methods in terms of influence. We study the effect of the community structure on the performance of the proposed framework. Our experiments show that the community structures with higher 'modularity' (defined in Section II) lead the proposed framework to perform better in terms of run-time and influence.

D. Organization

The rest of the paper is organized as follows. In Section II, we discuss the preliminaries and formulate the problem. In Section III, we discuss our methodology. In Section IV, we discuss the experiments performed on real-world social networks. Section V concludes the paper and provides future directions.

II. Preliminaries and Problem Formulation

In this section, we discuss some preliminaries and formulate the problem of interest in this paper. Refer to Appendix A for a table of important notations used throughout the paper.

A. Submodularity

Let Ω denote the ground set of n elements and 2^{Ω} denote the set of all subsets of Ω. A set function $f: 2^{\Omega} \rightarrow \mathbb{R}$ is said to be *submodular* if it satisfies a natural 'diminishing returns' property: the marginal gain from adding an element v to a set $S \subseteq \Omega$ is at least as high as the marginal gain from adding

the same element v to a superset $T \subseteq \Omega$ of S. Formally, for any sets $S, T \subseteq \Omega$ such that $S \subseteq T$, f satisfies

$$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T). \tag{1}$$

A set function $f : 2^{\Omega} \rightarrow \mathbb{R}$, is said to be *monotone* (non-decreasing) if for any sets $S, T \subseteq \Omega$ such that $S \subseteq T$, f satisfies

$$f(S) \leq f(T). \tag{2}$$

B. Diffusion models and social influence

There are several discrete-time stochastic models of diffusion over social networks. For the purpose of our research, we focus on the *independent cascade* (IC) [5], [6] and *linear threshold* (LT) [7], [8] models of diffusion.

In the independent cascade model, given a graph $G = (V, E)$, the process starts at time 0 with an initial set of active nodes S, called the *seed set*. When a node $v \in S$ first becomes active at time t, it will be given a single chance to activate each currently inactive neighbor w, it succeeds with a probability $p_{v,w}$ (independent of the history thus far). If w has multiple newly activated neighbors, their attempts are sequenced in an arbitrary order. If v succeeds, then w will become active at time $t + 1$; but whether or not v succeeds, it cannot make any further attempts to activate w in subsequent rounds. The process runs until no further activation is possible.

In the linear threshold model, given a graph $G = (V, E)$, a node v is influenced by each neighbor w according to a weight $p_{v,w}$ such that $\sum_{w \in \partial v} p_{v,w} \leq 1$, where ∂v represents the set of neighbors of v. Each node v chooses a *threshold* θ_v uniformly from the interval [0,1]; this represents the weighted fraction of v's neighbors that must become active in order for v to become active. The process starts with a random choice of thresholds for the nodes, and an initial set of active nodes S, called the *seed set*. In step t, all nodes that were active in step $t - 1$ remain active, and we activate any node v for which the total weight of its active neighbors is at least θ_v. The process runs until no more activation is possible.

Note that both these processes of diffusion are *progressive*, i.e. the nodes can switch from being inactive to active, but do not switch in the other direction.

At any time t in the cascade, each node $v \in V$ can be either active or inactive. We denote the process for each node $v \in V$'s state as $\{Y_t^{(v)}\}_{t=1}^{T}$

$$Y_t^{(v)} = \begin{cases} 1, & \text{if node } v \text{ is active at time } t, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The *influence* $\sigma(S)$ of a set S is defined as the expected number of active nodes at the end of the cascade (denoted by time T), given that S is the set of initially active nodes,

$$\sigma(S) = \mathbb{E}\left[\sum_{v \in V} Y_T^{(v)} \middle| \bigcap_{v \in V} \{Y_0^{(v)} = \mathbb{1}(v \in S)\}\right], \tag{4}$$

where $\mathbb{1}(\cdot)$ denotes the indicator function.

Kempe et al. [3] showed that under common models of diffusion such as independent cascade and linear threshold models, $\sigma(S)$ is a monotone non-decreasing submodular set function.

C. Problem statement

For a given integer budget k, we are interested in finding a k-node subset of the set of nodes V, which has the maximum influence over all possible k-node subsets of V. Formally, the problem of influence maximization (IM) is defined as

Problem 1.

$$\begin{aligned} &\arg\max_{S \subseteq V} \ \sigma(S), \\ &\textit{s.t.} \quad |S| \leq k. \qquad \text{(budget constraint)} \end{aligned}$$

III. Methodology

With the goal of solving the influence maximization problem (Problem 1) in a time-efficient manner, we propose a community-aware divide-and-conquer framework. The proposed framework reduces the search space for the seed sets by partitioning the given network using its inherent community structure. The proposed framework involves (i) learning the inherent community structure of the social network, (ii) generating candidate solutions by solving the influence maximization problem for each community, and (iii) selecting the final set of individuals to be incentivized from the candidate solutions using a novel progressive budgeting scheme.

Algorithm 1 outlines the framework proposed in this paper. It uses three sub-routines which are explained in the following subsections.

Algorithm 1 Community-IM

1: **Input** Graph G, budget k, `com-method`, `sol-method`.
2: $\{G_i\}_{i=1}^{c} \leftarrow$ Community-Detection(G, `com-method`)
3: **for** community $i = 1, \ldots, c$ **do**
4: $\quad \mathcal{S}_i, \Sigma_i \leftarrow$ Generate-Candidates(G_i, k, `sol-method`)
5: **end for**
6: $S^* \leftarrow$ Progressive-Budgeting($\{\mathcal{S}_i\}_{i=1}^{c}$,$\{\Sigma_i\}_{i=1}^{c}$,$k$)
7: **return** S^*

A. Learning the community structure of the network

For the given social network $G = (V, E)$, we obtain a hard partition $\{V_1, \ldots, V_c\}$ of the node set V using some community detection method. By hard partitioning, we mean we mean the communities are non-overlapping, i.e. $V_i \cap V_j = \emptyset$ for all communities $i \neq j$ with $i, j \in \{1, \ldots, c\}$ and $\bigcup_{i=1}^{c} V_i = V$. Define $G_i = (V_i, E_i)$ where E_i is the set of edges from E connecting pairs of nodes in V_i. We call $\{G_1, \ldots, G_c\}$ a network-partition.

Most community detection methods select communities such that the nodes within a community are more 'well-connected' than the nodes between communities. Methods differ in how they explicitly or implicitly measure the connectedness of the nodes in a network. Common community detection methods are the Louvain method [32], label propagation [33], and the Girvan-Newman algorithm [34].

1) Quality of a network-partition: The quality of a network-partition can be measured using modularity score [35], [36]. The modularity score of a network-partition is defined as the fraction of the edges that fall within the given groups minus the expected fraction if edges were distributed at random. For a network-partition $\{G_1, \ldots, G_c\}$, modularity [36] is defined as

$$Q = \sum_{i=1}^{c} \left[\frac{L_i}{|E|} - \left(\frac{\delta_i}{|E|} \right)^2 \right],$$

where L_i is the number of edges between the pairs of nodes in G_i and δ_i is the sum of the degrees of nodes in G_i.

The modularity score is used as a measure of how well a community detection algorithm partitions a network. A higher value of modularity corresponds to a network-partition with higher connectedness within each community.

2) Community detection methods: We discuss some commonly used community detection methods (`com-method` in Algorithm 1). The Louvain method [32] first obtains small communities by optimizing modularity locally on all of the nodes. Then each small community is treated as a single node and the previous step is repeated. Label propagation [33] starts with a (generally small) random subset of the nodes with community labels. The algorithm then iteratively assigns labels to previously unlabeled nodes. The Girvan-Newman method [34] method uses a measure known as 'betweenness.' Define the betweenness of an edge [34] as the sum of the 'weights' of the shortest paths between any pair of nodes that run along it. If there are d different shortest paths between any two nodes then the weight of each path is set as $1/d$. The Girvan-Newman method [34] method involves the following steps.

1) First, calculate the betweenness of all existing edges in the network.
2) Next, remove the edge(s) with the highest betweenness.
3) Finally, recalculate the betweenness of all edges affected by the removal at the previous step.
4) Repeat the previous two steps until no edge remains.

For the framework proposed in this paper, the only formal requirement for the community detection method (`com-method` in Algorithm 1) is that it provides a hard partition. Based on our experiments (discussed in Section IV), we observe that the Louvain method [32] works the best for our framework.

B. Generating candidate solutions by solving the influence maximization problem for each community

For each community G_i, we find the best seed sets of sizes $1, \ldots, k$ for that community using some standard influence maximization method. Let $S_{i,j}$ denote the best seed set of size j for community i. Let $\sigma_i(S_{i,j})$ denote the corresponding expected influences of those seed sets within community i ($i = 1, \ldots, c$).

Solving the influence maximization problem separately for different communities instead of the entire network improves the empirical run-time. The partitioning of the original network leads to fewer subset evaluations (oracle calls). Furthermore, those (fewer) evaluations are also faster to carry out. Refer to Appendix E-B for details.

For the framework proposed in this paper, any standard influence maximization method can be used as `sol-method` in Algorithm 1. For our experiments (discussed in Section IV), we use the CELF++ method [11] to demonstrate our framework.

Later, to discuss guarantees of our method (on a surrogate optimization problem), we will assume that the `sol-method` used has the following property.

Assumption 1. *We assume that the marginal gains $\{\sigma_i(S_{i,j+1}) - \sigma_i(S_{i,j})\}_{j=1}^{k-1}$ within each community $i \in \{1, \ldots, c\}$ are non-increasing.*

Remark 1. *Assumption 1 will automatically hold if the solutions are nested (i.e. $S_{i,j} \subset S_{i,j+1}$) due to submodularity. Iterative greedy influence maximization methods, such as those based on the [9], return nested solutions by design. Assumption 1 also holds automatically for optimal subsets regardless of nesting (due to submodularity), though it is computationally prohibitive to identify optimal subsets.*

C. Selecting the final seed set

After separately solving the influence maximization problem for each community, we allocate the total budget k across the c communities based on the within-community influences $\{\sigma_i(S_{i,j}) | i \in \{1, \ldots, c\}, j \in \{1, \ldots, k\}\}$.Formally, we solve the binary *integer linear program* (ILP) described as Problem 2.

Problem 2.

$$\begin{aligned}
&\underset{\substack{\{x_{i,j}\}\\ i=1,\ldots,c\\ j=1,\ldots,k}}{\arg\max} \sum_{i=1}^{c}\sum_{j=1}^{k} x_{i,j}\sigma_i(S_{i,j}),\\
\textit{s.t.}\quad &\sum_{i=1}^{c}\sum_{j=1}^{k} x_{i,j}|S_{i,j}| \le k, && \text{(budget constraint)}\\
&\sum_{j=1}^{k} x_{i,j} \le 1 \ \forall i = 1, \ldots, c, && \text{(no repetition)}\\
&x_{i,j} \in \{0,1\} \ \forall i, j. && \text{(binary integer constraints)}
\end{aligned}$$

Before discussing how we propose to solve Problem 2, we first discuss how we use the solution to this ILP for selecting a seed set and how the objective functions of Problems 1 and 2 relate.

Let x^* denote the optimal solution to Problem 2. If there are multiple optimal solutions pick one arbitrarily. Denote the budget allocated to each community i as k_i (e.g. the index j for which $x^*_{i,j} = 1$). We next construct a seed set for Problem 1 based on the allocation budget x^*,

$$S^* \leftarrow \bigcup_{i=1}^{c} S_{i,k_i}. \tag{5}$$

The objective function in Problem 2 lower bounds the objective function of Problem 1, with equality if G is formed of disjoint communities.

Theorem 1. *Consider any network partition $\{G_i\}_{i=1}^c$ of G and any set of subsets $\{S_i\}_{i=1}^c$ of nodes such that $S_i \subseteq V_i$ for $i = 1, \ldots, c$. Then*

$$\sum_{i=1}^{c} \sigma_i(S_i) \leq \sigma(\cup_{i=1}^c S_i).$$

The proof is in Appendix C.

In general, solving an ILP is an NP-Complete problem [37]. However, the submodularity of the influence allows us to solve Problem 2 in polynomial time.

1) Progressive Budgeting: By Assumption 1 (by submodularity for nested subsets), we know that the marginal gain in influence due to each additional node in the seed set is diminishing (both for each individual community and overall since sums of submodular functions are submodular). Hence, we can progressively allocate the budget across the community-based seed sets $\{S_{i,j}\}$. The Progressive-Budgeting sub-routine used in Algorithm 1 is outlined in Algorithm 2.

Algorithm 2 Progressive-Budgeting

Input $\mathcal{S}, \Sigma, k$.
$\{S_{i,j} | i \in \{1, \ldots, c\}, j \in \{1, \ldots, k\}\} \leftarrow \mathcal{S}$
$\{\sigma_i(S_{i,j}) | i \in \{1, \ldots, c\}, j \in \{1, \ldots, k\}\} \leftarrow \Sigma$
$\{\delta_i\}_{i=1}^c \leftarrow \{\sigma_i(S_{i,1})\}_{i=1}^c$ ▷ Initialize the marginal gains.
$\{k_i\}_{i=1}^c \leftarrow \{0\}_{i=1}^c \ \forall i$ ▷ Initialize the budget allocations.
$S^* \leftarrow \emptyset$ ▷ Initialize the final set.
for $\ell = 1, \ldots, k$ **do**
    $m \leftarrow \arg\max_{i \in \{1,\ldots,c\}} \delta_i$ ▷ Index of the community with the maximum marginal gain.
    $k_m \leftarrow k_m + 1$ ▷ Update the budget allocated to community m.
    $\delta_m \leftarrow \sigma_m(S_{m,k_m+1}) - \sigma_m(S_{m,k_m})$ ▷ Update the marginal gains for community m.
end for
$S^* \leftarrow \bigcup_{i=1}^c S_{i,k_i}$
return S^* ▷ Final seed set.

An illustrative example of progressive budgeting is provided in Appendix B. We will next discuss the correctness of Algorithm 2. The correctness of Algorithm 2 will follow from the following lemma, asserting that up to the uniqueness of optimal solutions of Problem 2 for different cardinalities, the optimal budget allocations are nested.

For each budget $\ell \in \{1, \ldots, k\}$, let $S^{*,(\ell)}$ denote an optimal seed set (5) of cardinality ℓ and let $\mathbf{k}^{*,(\ell)} = \{k_i\}_{i=1}^c$ denote the budget allocations to the c communities. We say a sequence $(S^{*,(\ell)})_{\ell=1}^k$ of (optimal) seed sets is nested if the seed sets are proper subsets of each other (i.e. $S^{*,(\ell)} \subset S^{*,(\ell+1)}$). We say a sequence $(\mathbf{k}^{*,(\ell)})_{\ell=1}^k$ of budget allocations is nested if across the sequence each community's allocation is non-decreasing (i.e. $k_i^{*,(\ell)} \leq k_i^{*,(\ell+1)}$).

Lemma 1. *Under Assumption 1, there is a nested sequence $\{\mathbf{k}^{*,(\ell)}\}_{\ell=1}^k$ of optimal budget allocations for Problem 2.*

The proof is in Appendix D.

Theorem 2. *Under Assumption 1, Algorithm 2 solves Problem 2.*

The proof follows immediately from Lemma 1 and the greedy design of Algorithm 2.

Remark 2. *In general, the guarantees of Theorem 2 do not translate into guarantees for Problem 1. Since Problem 1 is an NP-hard problem for common diffusion models on a general network, common methods are approximation algorithms (with an approximation ratio of $(1 - 1/e)$ or slightly worse) or heuristics. Thus, the inputs to Algorithm 2 in general will not necessarily be optimal seed sets for their respective communities. Additionally, as noted in Theorem 1, the objective functions in Problems 1 and 2 only match if the original network G is disjoint (and the communities selected align with the segments of G).*

The computational complexity of the proposed framework (Algorithm 1) is analyzed in Appendix E.

IV. Experiments

We evaluated the performance of our framework using real-world social networks. We next discuss the network data used for our experiments, list the algorithms chosen for comparison, provide experimental details, and then present results and discussion.

A. Network data

We used 4 real-world social networks for our experiments. The data is available at Stanford Large Network Dataset Collection [38]. The number of nodes, number of edges, and modularity (for the network-partition obtained using the Louvain method [32]) of each network are provided in Table I.

TABLE I: Basic information of the networks used.

Network	Nodes	Edges	Modularity
Facebook [39]	4,039	88,234	0.8678
Bitcoin [40], [41]	5,881	35,592	0.4196
Wikipedia [42], [43]	7,115	103,689	0.4175
Epinions [44]	75,879	508,837	0.8219

The Facebook network [39] consists of a dataset consisting of 'circles' (or 'friends lists') from Facebook. The Facebook network is undirected; we converted it to a directed network by replacing each edge with two directed edges. Bitcoin network [40], [41] is a (directed) who-trusts-whom network of people who trade using Bitcoin on a platform called Bitcoin OTC. Wikipedia network [42], [43] is a who-votes-on-whom (directed) network to become an administrator. Epinions network [44] is a who-trust-whom (directed) online social network of a general consumer review platform called Epinions.

For edge-weights, two models are used which are *weighted cascade* (WC) model [3] where for each node $v \in V$, the weight of each edge entering v was set to $1/\text{in-degree}(v)$ and *trivalency* (TV) model [22] where each edge-weight was drawn uniformly at random from a small set of constants $\{0.1, 0.01, 0.001\}$. However, for the linear threshold model (LT) of diffusion, only the WC model is used for edge-weights as the TV model does not necessarily maintain the sum of weights of all edges incident on a node to be less than or equal to 1.

B. Algorithms

We compared the proposed community-aware framework (Community-IM) with the following algorithms.

1) CELF++ [11], the state-of-the-art simulation-based greedy algorithm.
2) CoFIM [20], a community-aware heuristic algorithm with guarantees under the independent cascade diffusion model with the weighted-cascade edge-weight model.
3) DSGA [13], a recent genetic algorithm-based method that uses Monte Carlo simulations.
4) Degree [3], the simplest heuristic algorithm where for budget k, top-k out-degree nodes are selected.
5) Degree-Discount [4], a modification of the the Degree heuristic algorithm with better empirical performance.

Note that the CoFIM algorithm was developed only for IC diffusion model with WC edge-weight model. However, for empirical comparisons, we implemented it for the other choices of diffusion models and edge-weight models as well.

For the purpose of demonstrating the performance of the proposed framework, Community-IM (Algorithm 1), we used the Louvain method [32] as `com-method`, and CELF++ [11] as `sol-method` for Community-Detection and Generate-Candidates subroutines, respectively. In general, the user may try different combinations of `com-method` and `sol-method` as part of the proposed framework.

We also studied the effect of the (modularity of) the community structure on the performance of the proposed framework. We used the Louvain [32], Label Propagation [33], and Girvan-Newman [34] community-detection methods (discussed in Section III-A2) as `com-method` in Community-Detection step of the proposed framework. For brevity, we only considered the Facebook network under different diffusion models and WC edge-weight model.

C. Experimental details

We used the budgets $k = 1, 5, 10, \ldots, 100$ for comparing different algorithms. However, for DSGA [13], we only used the budgets $k = 1, 20, 40, \ldots, 100$ due to its high run-time. For brevity, for studying the effect of the community structure on the proposed framework, we used the budgets $k = 1, 5, \ldots, 50$. The influence of any seed set was estimated as the average number of active nodes from $1,000$ different Monte Carlo simulations of the underlying diffusion starting with the same seed set. For any network, if a community detection method returned some communities whose individual sizes are below 1% of the number of nodes in the network then we merged them all into a single community. We do this to avoid having too many small communities.

The experiments were carried out on a computer with 2.6 GHz 24-core Intel Xeon Gold Sky Lake processors and 96 GB of memory. We used Python for our implementation. The source codes of CELF++ and CoFIM provided by their authors are written in C++. The data and source code for this paper are available here.

D. Results

For different networks under different diffusion models and edge-weight models,

- Figures 1-3 show the influences of chosen seed sets using different algorithms for different values of budget k. Figure 1 shows the results for IC diffusion model and WC edge-weight model, Figure 2 shows the results for IC diffusion model and TV edge-weight model, and Figure 3 shows the results for LT diffusion model and WC edge-weight model.
- Table II and Table III show the influences and run-times, respectively for budget $k = 100$ for different algorithms.

For the Facebook network under different diffusion models, and WC edge-weight model for different community detection methods as `com-method` in Community-Detection step of the proposed framework,

- Figure 4 shows the influences of chosen seed sets using different algorithms for different values of k.
- Table IV shows the modularity scores, the number of communities, and the influences and run-times for budget $k = 50$ for Community-IM and CELF++.

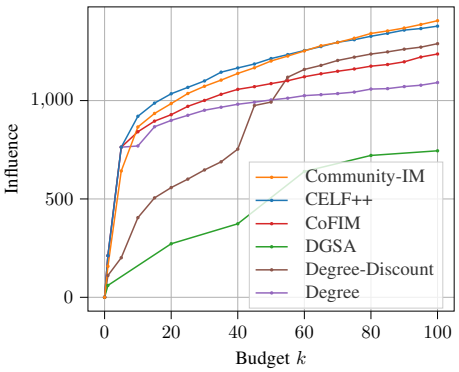
E. Discussion

1) Overview: The proposed framework (Community-IM) achieves either marginally lower, equal, or higher influence compared to CELF++, and achieves better influence compared to all other algorithms. This performance in terms of influence improves as the budget increases. The proposed framework brings savings in terms of run-time as compared to the simulation-based methods. The community structures with higher modularity lead the proposed framework to perform better in terms of run-time and influence. Moreover, these observations vary across different networks, diffusion models, edge-weight models, and budgets.

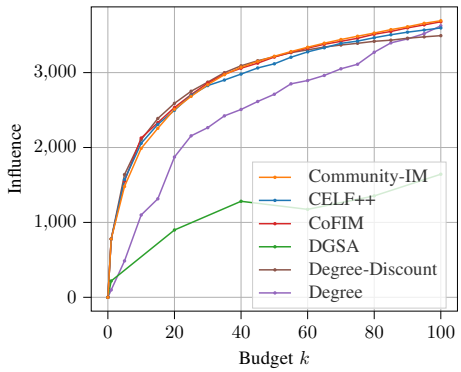
2) Performance in terms of influence: For low budgets, the influence for Community-IM (orange) is marginally lower than that for CELF++ (blue). However, for high budgets, the influence for Community-IM is the same or higher than that for CELF++. Furthermore, the influence for Community-IM is higher compared to the rest of the algorithms. We observe this trend for the Facebook, Bitcoin, and Epinions networks under different diffusion models and different edge-weight models from Figures 1(a), 1(b), and 1(d), Figures 2(a) and 2(b), and Figures 3(a) and 3(b).

For all budgets, the influence for Community-IM (orange) is marginally lower than that for CELF++ (blue). However, the gap between the influence for Community-IM and that for CELF++ decreases as the budget k increases. Furthermore, the influence for Community-IM is higher compared to the rest of the algorithms. We observe this trend for the Wikipedia network under different diffusion and edge-weight models from Figures 1(c), 2(c), and 3(c).

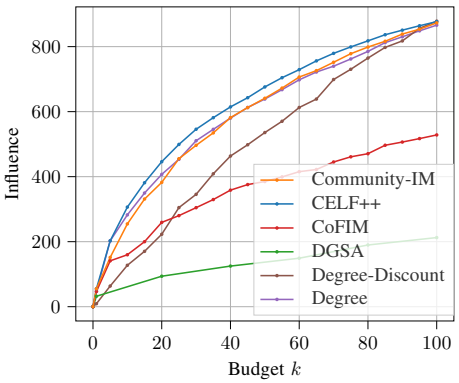
Note that by design, the proposed community-aware framework gives preference to the community-level influential nodes while building its nested solution using progressive budgeting (Algorithm 2). However, when the budget is large, depending



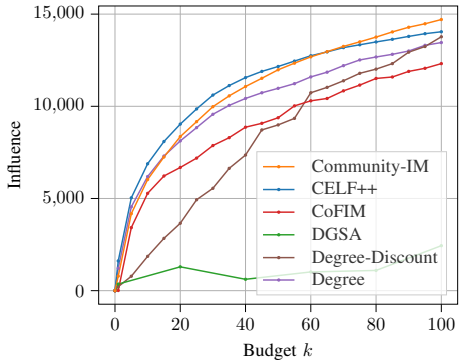
(a) Facebook network



(b) Bitcoin network

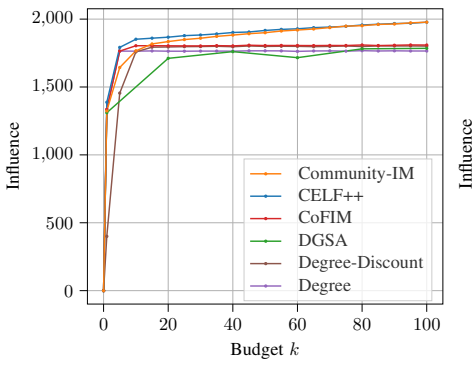


(c) Wikipedia network

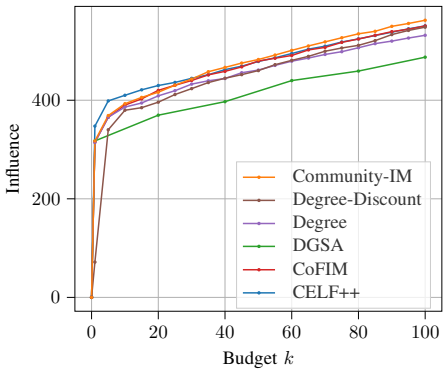


(d) Epinions network

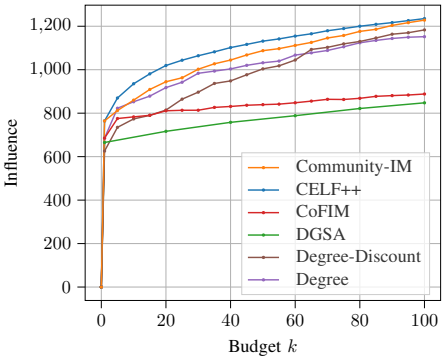
Fig. 1: Influence vs. budget k for different networks under IC diffusion model and WC edge-weight model.



(a) Facebook network



(b) Bitcoin network



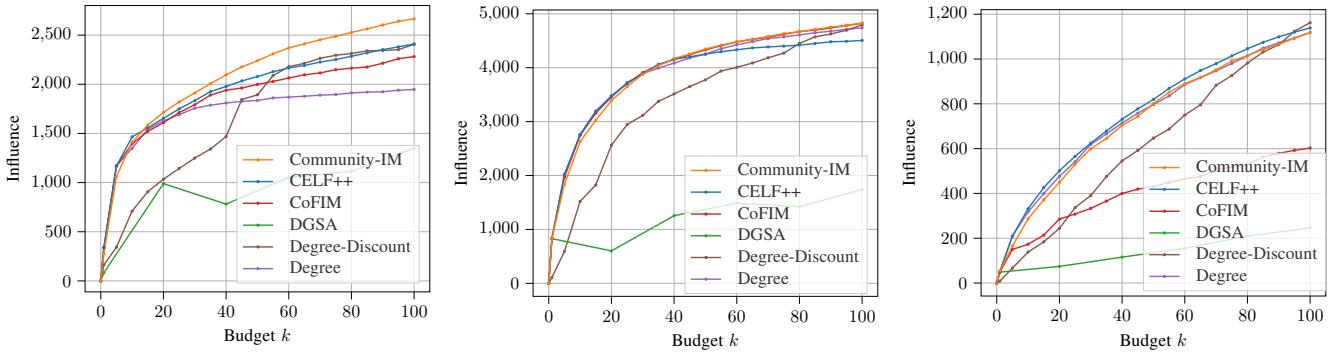
(c) Wikipedia network

Fig. 2: Influence vs. budget k for different networks under IC diffusion model and TV edge-weight model.

on their community-level influence, the network-level influential nodes are also selected. On the contrary, CELF++ prefers network-level influential nodes while building its nested solution. Hence, the proposed framework takes advantage of the community-level influence ordering of nodes early on. However, network-level celebrities may not be equally popular within each community. Hence, particularly for low budgets, the proposed framework selects only the community-level influential nodes. However, when the budget is large, it starts to pick the network-level influential nodes as well. This explains why the performance of the proposed algorithm in terms of influence gets better as the budget increases. Such a trend gets more pronounced for networks that have some extremely (network-level) influential nodes (e.g. the Facebook and Epinions networks) that are not selected initially for small values of the budget but included later for high budgets.

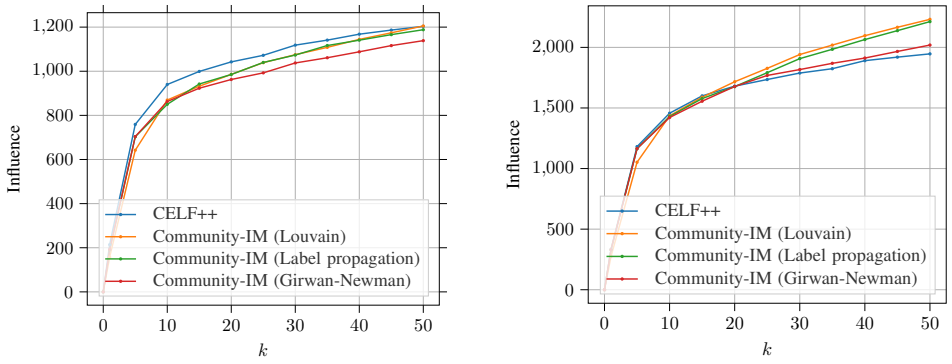
Moreover, Table II shows that for each network, the influence of the chosen seed set of size 100 using Community-IM is close to or even better than the same for CELF++ under different diffusion models and edge-weight models.

3) Performance in terms of run-time: Table III shows that the proposed framework brings savings in terms of run-time as compared to the simulation-based methods (CELF++, and DSGA) across different networks, diffusion models, and edge-



(a) Facebook network (b) Bitcoin network (c) Wikipedia network

Fig. 3: Influence vs. budget k for different networks under LT diffusion model and WC edge-weight model.



(a) Independent cascade (IC) diffusion model (b) Linear threshold (LT) diffusion model

Fig. 4: Influence vs. budget k for the Facebook network under different diffusion models and WC edge-weight model.

TABLE II: Comparison of influences for budget $k = 100$.

Diffusion model	Edge-weight model	Network	Community-IM	CELF++	CoFIM	DSGA	Degree	Degree-Discount
Independent cascade	Weighted cascade	Facebook	1,378	1,406	1,237	846	1,092	1,289
		Bitcoin	3,693	3,493	3,679	1,643	3,596	3,625
		Wikipedia	873	877	528	213	866	878
		Epinions	14,706	14,043	12,315	2,439	13,458	13,771
	Trivalency	Facebook	1,977	1,977	1,809	1,305	1,765	1,801
		Bitcoin	562	551	551	487	532	548
		Wikipedia	1,228	1,235	888	848	1,152	1,183
Linear threshold	Weighted cascade	Facebook	2,231	1,946	1,936	969	1,835	2,000
		Bitcoin	4,829	4,506	4,822	1,743	4,740	4,794
		Wikipedia	1,117	1,139	602	246	1,119	1,162

weight models. Moreover, these run-time savings are more pronounced for larger networks. The gains in terms of run-time also vary across diffusion models and edge-weight models. We observe the highest gains for IC diffusion model with TV edge-weight model and the least gains for the IC diffusion model with WC edge-weight model.

4) Effect of the community structure on the performance of the proposed framework: Based on Figure 4, we observe that the community structures with higher values of modularity (obtained using the Louvain and Label Propagation methods) lead the proposed framework to do better in terms of influence as compared to the community structures with lower values of modularity (obtained using the Girvan-Newman method [34]). Furthermore, for all budgets, the influences for Community-IM with Louvain method and Community-IM with Label propagation method are close to each other which can be attributed to the fact that the modularity scores of the partitions obtained by these two methods are quite close.

Table IV shows that the influence for the budget of $k = 50$, using Community-IM is close to or even better than the same for CELF++ for different choices of community detection methods under different diffusion models and WC edge-weight model. Furthermore, the performance of Community-IM compared to CELF++ in terms of influence and run-time improves as the modularity of the partition and the number of communities increase. Note that, for the proposed framework, the Louvain method is the best choice of community detection method while the Girvan-Newman method performs the worst. The Louvain method partitions the graph into 18 communities with the largest community having 523 nodes (approximately

TABLE III: Comparison of run-times (in seconds) for budget $k = 100$.

Diffusion model	Edge-weight model	Network	Community-IM	CELF++	CoFIM	DSGA
Independent cascade	Weighted cascade	Facebook	3,782	17,359	547	9,1267
		Bitcoin	850	10,859	35	20,825
		Wikipedia	3,477	2,660	213	18,447
		Epinions	16,465	250,241	7,397	267,796
	Trivalency	Facebook	7,195	74,684	567	312,948
		Bitcoin	576	7,818	35	34,453
Linear threshold	Weighted cascade	Facebook	8,545	46,771	554	65,391
		Bitcoin	1,077	45,747	36	62,184
		Wikipedia	4,628	5,940	224	23,307

TABLE IV: Comparison of influences and run-times (in seconds) for budget $k = 50$ for the Facebook network under WC edge-weight model for different community detection methods.

				Influence		Run-times (in seconds)	
Diffusion model	Community detection method	No. of communities	Modularity score	Community-IM	CELF++	Community-IM	CELF++
Independent cascade	Louvain	18	0.8678	1,205	1,203	3,069	14,077
	Label propagation	11	0.7368	1,188	1,203	4,068	14,077
	Girvan-Newman	2	0.0439	1,139	1,203	14,221	14,077
Linear threshold	Louvain	18	0.8304	2,231	1,946	7,224	38,968
	Label propagation	11	0.7368	2,213	1,946	12,961	38,968
	Girvan-Newman	2	0.0439	2,019	1,946	34,606	38,968

10% of the size of the entire network). Hence, Community-IM does not come across any giant component (causing lengthier diffusions) while estimating the within-community influence. Contrary to this, the Girvan-Newman method partitions the network into just two communities with the largest community having 3,833 nodes (very close to the size of the entire network). This makes the within-community diffusions take longer to finish while using the communities obtained using the Girvan-Newman method. This explains why Community-IM with the Girvan-Newman method runs slower as compared to the same with the Louvain method.

V. Conclusion and Future Work

For solving the problem of influence maximization on social networks, we leveraged the inherent community structure of a network and proposed a novel community-aware framework for maximizing the spread of influence through a social network in a fast manner. Based on our experiments, we conclude that the proposed framework outperforms the standard simulation-based methods in terms of run-time and the heuristic methods in terms of influence. As the proposed method leverages the inherent community structure of the network, we also studied the effect of the community structure on the performance of our framework. Based on our experiments, we conclude that the community structures with higher modularity lead the proposed framework to perform better in terms of run-time and influence. Among the methods considered in this paper, we find the Louvain method [32] works best for our framework.

We point out two limitations of our method. First, our method requires the communities learned during Step 1 to be non-overlapping. However, in general, a social network may have overlapping communities. Second, our method does not explicitly account for the inter-community influence while generating the candidate solutions during Step 2. In the future, we want to extend our method to handle overlapping community structures and explicitly account for the inter-community influence. Other future directions are to extend the proposed community-aware framework to competitive influence maximization [45], data-based influence maximization [22], and full-bandit online influence maximization [30], [31].

References

[1] D. Evans, *Social Media Marketing: The Next Generation of Business Engagement*. John Wiley & Sons, 2010.

[2] P. Domingos and M. Richardson, "Mining the network value of customers," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2001, pp. 57–66.

[3] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2003, pp. 137–146.

[4] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009, pp. 199–208.

[5] J. Goldenberg, B. Libai, and E. Muller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," *Marketing Letters*, vol. 12, no. 3, pp. 211–223, 2001.

[6] ——, "Using complex systems analysis to advance marketing theory development: Modeling heterogeneity effects on new product growth through stochastic cellular automata," *Academy of Marketing Science Review*, vol. 9, no. 3, pp. 1–18, 2001.

[7] M. Granovetter, "Threshold models of collective behavior," *American Journal of Sociology*, vol. 83, no. 6, pp. 1420–1443, 1978.

[8] T. C. Schelling, *Micromotives and Macrobehavior*. WW Norton & Company, 2006.

[9] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—I," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[10] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007, pp. 420–429.

[11] A. Goyal, W. Lu, and L. V. Lakshmanan, "Celf++: Optimizing the greedy algorithm for influence maximization in social networks," in *Proceedings of the 20th International Conference Companion on World Wide Web*, 2011, pp. 47–48.

[12] C. Borgs, M. Brautbar, J. Chayes, and B. Lucier, "Maximizing social influence in nearly optimal time," in *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2014, pp. 946–957.

[13] J. J. Lotf, M. A. Azgomi, and M. R. E. Dishabi, "An improved influence maximization method for social networks based on genetic algorithm," *Physica A: Statistical Mechanics and its Applications*, vol. 586, p. 126480, 2022.

[14] Y.-C. Chen, W.-C. Peng, and S.-Y. Lee, "Efficient algorithms for influence maximization in social networks," *Knowledge and Information Systems*, vol. 33, no. 3, pp. 577–601, 2012.

[15] H. Ma, H. Yang, M. R. Lyu, and I. King, "Mining social networks using heat diffusion processes for marketing candidates selection," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, 2008, pp. 233–242.

[16] Y.-C. Chen, W.-Y. Zhu, W.-C. Peng, W.-C. Lee, and S.-Y. Lee, "CIM: Community-based influence maximization in social networks," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 2, pp. 1–31, 2014.

[17] A. Bozorgi, H. Haghighi, M. S. Zahedi, and M. Rezvani, "INCIM: A community-based algorithm for influence maximization problem under the linear threshold model," *Information Processing & Management*, vol. 52, no. 6, pp. 1188–1199, 2016.

[18] A. Bozorgi, S. Samet, J. Kwisthout, and T. Wareham, "Community-based influence maximization in social networks under a competitive linear threshold model," *Knowledge-Based Systems*, vol. 134, pp. 149–158, 2017.

[19] S. Bharathi, D. Kempe, and M. Salek, "Competitive influence maximization in social networks," in *Internet and Network Economics*, X. Deng and F. C. Graham, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 306–311.

[20] J. Shang, S. Zhou, X. Li, L. Liu, and H. Wu, "CoFIM: A community-based framework for influence maximization on large-scale networks," *Knowledge-Based Systems*, vol. 117, pp. 88–100, 2017.

[21] H. Huang, H. Shen, Z. Meng, H. Chang, and H. He, "Community-based influence maximization for viral marketing," *Applied Intelligence*, vol. 49, no. 6, pp. 2137–2150, 2019.

[22] A. Goyal, F. Bonchi, and L. V. Lakshmanan, "A data-based approach to social influence maximization," *Proceedings of the VLDB Endowment*, vol. 5, no. 1, pp. 73–84, 2011.

[23] Y. Pan, X. Deng, and H. Shen, "Credit distribution for influence maximization in online social networks with time constraint," in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*. IEEE, 2015, pp. 255–260.

[24] X. Deng, Y. Pan, Y. Wu, and J. Gui, "Credit distribution and influence maximization in online social networks using node features," in *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*. IEEE, 2015, pp. 2093–2100.

[25] S. Lei, S. Maniu, L. Mo, R. Cheng, and P. Senellart, "Online influence maximization," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015, pp. 645–654.

[26] Z. Wen, B. Kveton, M. Valko, and S. Vaswani, "Online influence maximization under independent cascade model with semi-bandit feedback," in *Advances in Neural Information Processing Systems*, 2017, pp. 3022–3032.

[27] S. Vaswani, B. Kveton, Z. Wen, M. Ghavamzadeh, L. Lakshmanan, and M. Schmidt, "Diffusion independent semi-bandit influence maximization," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017.

[28] S. Li, F. Kong, K. Tang, Q. Li, and W. Chen, "Online influence maximization under linear threshold model," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1192–1204, 2020.

[29] P. Perrault, J. Healey, Z. Wen, and M. Valko, "Budgeted online influence maximization," in *International Conference on Machine Learning*. PMLR, 2020, pp. 7620–7631.

[30] M. Agarwal, V. Aggarwal, A. K. Umrawal, and C. J. Quinn, "Stochastic top k-subset bandits with linear space and non-linear feedback with applications to social influence maximization," *ACM/IMS Transactions on Data Science (TDS)*, vol. 2, no. 4, pp. 1–39, 2022.

[31] G. Nie, M. Agarwal, A. K. Umrawal, V. Aggarwal, and C. J. Quinn, "An explore-then-commit algorithm for submodular maximization under full-bandit feedback," in *The 38th Conference on Uncertainty in Artificial Intelligence*, 2022.

[32] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.

[33] G. Cordasco and L. Gargano, "Community detection via semi-synchronous label propagation algorithms," in *2010 IEEE International Workshop on: Business Applications of Social Network Analysis (BASNA)*. IEEE, 2010, pp. 1–8.

[34] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[35] M. Newman, *Networks*. Oxford University Press, 2018.

[36] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, p. 066111, 2004.

[37] R. Kannan and C. L. Monma, "On the computational complexity of integer programming problems," in *Optimization and Operations Research*. Springer, 1978, pp. 161–172.

[38] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[39] J. Leskovec and J. J. Mcauley, "Learning to discover social circles in ego networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 539–547.

[40] S. Kumar, F. Spezzano, V. Subrahmanian, and C. Faloutsos, "Edge weight prediction in weighted signed networks," in *Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE, 2016, pp. 221–230.

[41] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian, "Rev2: Fraudulent user prediction in rating platforms," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 2018, pp. 333–341.

[42] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2010, pp. 1361–1370.

[43] ——, "Predicting positive and negative links in online social networks," in *Proceedings of the 19th International Conference on World Wide Web*, 2010, pp. 641–650.

[44] M. Richardson, R. Agrawal, and P. Domingos, "Trust management for the semantic web," in *International Semantic Web Conference*. Springer, 2003, pp. 351–368.

[45] S. Bharathi, D. Kempe, and M. Salek, "Competitive influence maximization in social networks," in *International Workshop on Web and Internet Economics*. Springer, 2007, pp. 306–311.

[46] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," 2002, Technical Report.

Abhishek K. Umrawal is currently a Ph.D. Candidate in the School of Industrial Engineering at Purdue University, West Lafayette, IN 47907, USA, and a Visiting Lecturer in the Department of Computer Science and Electrical Engineering at the University of Maryland, Baltimore County, MD 21250, USA. He received an M.Sc. degree in Statistics from the Indian Institute of Technology Kanpur, India, and an M.S. degree in Economics from Purdue University. His research interests include causality, reinforcement learning, optimization, and network science with applications to social networks and intelligent transportation.

Christopher J. Quinn (Member, IEEE) is currently an Assistant Professor in the Department of Computer Science at Iowa State University, Ames, IA 50011, USA. He received a B.S. degree in Engineering Physics from Cornell University, and M.S. and Ph.D. degrees in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign. His current research interests include machine learning, information theory, and network science, with applications to neuroscience and social networks.

Vaneet Aggarwal (Senior Member, IEEE) is currently a Full Professor at Purdue University, West Lafayette, IN 47907, USA. He received a B.Tech. degree from the Indian Institute of Technology Kanpur, India, and M.A. and Ph.D. degrees from Princeton University, all in Electrical Engineering. His current research interests include machine learning and its applications in networking, transportation, and quantum systems.

APPENDIX A
TABLE OF NOTATIONS

TABLE V: Table of notations.

Symbol	Explanation
Ω	Ground set.
2^{Ω}	Set of all subsets of Ω.
$G = (V, E)$	Directed graph.
$V = (v_1, \ldots, v_n)$	Set of vertices or nodes.
n	Size of V.
$E = (e_1, \ldots, e_n)$	Set of directed edges where $e_i, i = 1, \ldots, n$ is are ordered pairs of nodes.
$p_{v,w}$	Weight of the edge $v \to w$.
∂v	Set of neighbors of node v.
$Y_t^{(v)}$	Activation/state of node v at time t.
k	Budget.
$\sigma(S)$	Influence of a set S of nodes.
c	Number of communities.
`com-method`	Community detection method.
`sol-method`	Influence maximization method.
$\{G_1, \ldots G_c\}$	A partition of G with c sub-graphs that are $G_1, \ldots G_c$.
$\{V_1, \ldots V_c\}$	Set of sets of vertices for all sub-graphs in the partition $\{G_1, \ldots G_c\}$.
n_i	Size of $V_i, i = 1, \ldots, c$.
$\{E_1, \ldots E_c\}$	Set of sets of edges for all sub-graphs in the partition $\{G_1, \ldots G_c\}$.
Q	Modularity of a network partition.
$S_{i,j}$	Best seed set of size j ($j = 1, \ldots, k$) from community i ($i = 1, \ldots, c$).
$\sigma_i(S_{i,j})$	Influence of $S_{i,j}$ within community i ($i = 1, \ldots, c$).
$\mathcal{S}_i$	Set of all candidate solutions from community $i = \{S_{i,j} : j = 1, \ldots, k\}$.
Σ_i	Influences of all candidate solutions from community $i = \{\sigma_i(S_{i,j}) : j = 1, \ldots, k\}$.
$\mathcal{S}$	Set of sets of all candidate solutions from all communities = $\{\mathcal{S}_i : i = 1, \ldots, c\}$.
Σ	Set of sets of influences of all candidate solutions from all communities = $\{\Sigma_i : i = 1, \ldots, c\}$.
S^*	Final solution using the proposed framework.

APPENDIX B
AN ILLUSTRATIVE EXAMPLE OF PROGRESSIVE BUDGETING

In this section, we provide an illustrative example of progressive budgeting. After executing the Community-Detection and the Generate-Candidates steps of the proposed framework, we obtain the following output.

$$S_{i,j} = \text{Candidate set of size } j \text{ from community } i,$$
$$\sigma_{i,j} := \sigma_i(S_{i,j}) = \text{Influence of } S_{i,j} \text{ within community } i,$$
$$i = 1, \ldots, j = 1, \ldots, k.$$

Let the budget, $k = 4$. No. of communities, $c = 5$. The influences of different candidate sets within different communities are given in Table VI(a). For every $i = 1, \ldots, c; j = 1, \ldots, k$, we calculate the marginal influences as $m_{i,j} := \sigma_i(S_{i,j}) - \sigma_i(S_{i,j-1})$, where $\sigma_i(S_{i,0}) = 0, \forall i$. The marginal influences for the influences given in Table VI(a) are provided in Table VI(b).

Influence

Community i	$\sigma_{i,1}$	$\sigma_{i,2}$	$\sigma_{i,3}$	$\sigma_{i,4}$
1	8	14	18	21
2	5	10	14	15
3	9	14	16	17
4	7	12	16	18
5	5	9	11	11

(a) Influences of candidate sets after step (ii).

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1	8	6	4	3
2	5	5	4	1
3	9	5	2	1
4	7	5	4	2
5	5	4	2	0

(b) Marginal Influences of candidate sets after step (ii).

TABLE VI: An example input for progressive budgeting.

The progressive budgeting scheme for the example in Table VI is explained in Table VII. At any iteration, the circled cells ($\textcircled{\cdot}$) are the ones whose maximum is to be obtained. An asterisk ($*$) is placed before the maximum value at the current iteration. The superscript(s) on any community label represents the nodes selected from that community (ordered based on the ordering in the corresponding candidate set obtained in step (ii)).

For the example we considered, the final seed set is $\{1^{1,2}, 3^1, 4^1\}$ which is equivalent to $S_{1,2} \cup S_{3,1} \cup S_{4,1}$.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1	⑧	6	4	3
2	⑤	5	4	1
3^1	$*$ ⑨	5	2	1
4	⑦	5	4	2
5	⑤	4	2	0

(a) Iteration 1: Allocating the first unit.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1^1	$*$ ⑧	6	4	3
2	⑤	5	4	1
3^1	9	⑤	2	1
4	⑦	5	4	2
5	⑤	4	2	0

(b) Iteration 2: Allocating the second unit.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
1^1	8	⑥	4	3
2	⑤	5	4	1
3^1	9	⑤	2	1
4^1	$*$ ⑦	5	4	2
5	⑤	4	2	0

(c) Iteration 3: Allocating the third unit.

Marginal Influence

Community i	$m_{i,1}$	$m_{i,2}$	$m_{i,3}$	$m_{i,4}$
$1^{1,2}$	8	$*$ ⑥	4	3
2	⑤	5	4	1
3^1	9	⑤	2	1
4^1	7	⑤	4	2
5	⑤	4	2	0

(d) Iteration 4: Allocating the fourth unit.

TABLE VII: An illustration of progressive budgeting.

APPENDIX C
PROOF OF THEOREM 1

We now prove Theorem 1.

Proof. The proof follows from the linearity of expectation in the definition of influence σ and monotonicity.

$$\begin{aligned}
\sigma(\cup_{i=1}^{c} S_i) &= \mathbb{E}\left[\sum_{v\in V} Y_T^{(v)} \middle| \bigcap_{v\in V}\{Y_0^{(v)} = \mathbb{1}(v \in \cup_{i=1}^{c} S_i)\}\right], && \text{(from (4))}\\
&= \sum_{i=1}^{c} \mathbb{E}\left[\sum_{v\in V_i} Y_T^{(v)} \middle| \bigcap_{v\in V}\{Y_0^{(v)} = \mathbb{1}(v \in \cup_{i=1}^{c} S_i)\}\right], && \text{(hard-partitioning; linearity of expectation)}\\
&\leq \sum_{i=1}^{c} \mathbb{E}\left[\sum_{v\in V_i} Y_T^{(v)} \middle| \bigcap_{v\in V}\{Y_0^{(v)} = \mathbb{1}(v \in S_i)\}\right], && (\sigma_i \text{ is monotone non-decreasing})\\
&= \sum_{i=1}^{c} \sigma_i(S_i).
\end{aligned}$$

□

APPENDIX D
PROOF OF LEMMA 1

We now prove Lemma 1.

Proof. The proof will follow by contradiction. Suppose there is some instance of Problem 2 and some cardinality $1 \leq \ell < k$ such that starting with the budget allocation $\mathbf{k}^{*,(\ell+1)}$ for the optimal solution $S^{*,(\ell+1)}$ for budget $\ell+1$ and removing a unit of budget from any community results in a sub-optimal allocation (i.e. worse than $\mathbf{k}^{*,(\ell)}$). For the special case that optimal solutions are unique for each cardinality and `sol-method` returns nested subsets, this condition simplifies to $S^{*,(\ell)} \not\subset S^{*,(\ell+1)}$.

If there is more than one optimal solution for budget ℓ, fix any one as $S^{*,(\ell)}$. Let us modify the budget allocation $\mathbf{k}^{*,(\ell+1)}$ of the optimal solution $S^{*,(\ell+1)}$ as follows. Pick any community $\tilde{i} \in \{1,\ldots,c\}$ such that

$$k_{\tilde{i}}^{*,(\ell)} \leq k_{\tilde{i}}^{*,(\ell+1)} - 1, \tag{6}$$

that is even after removing a unit of budget for community $\tilde{i}$ from allocation $\mathbf{k}^{*,(\ell+1)}$, there is still as much budget left over as there is for community $\tilde{i}$ in the allocation $\mathbf{k}^{*,(\ell)}$ (i.e. of the optimal solution $S^{*,(\ell)}$ for cardinality ℓ). Trivially since $\mathbf{k}^{*,(\ell+1)}$ allocates a larger budget overall, there must be one such community $\tilde{i}$ (if the solutions were nested, there would be exactly one). Denote the corresponding modified solution and budget allocation as $\tilde{S}^{*,(\ell+1)}$ and $\tilde{\mathbf{k}}^{*,(\ell+1)}$ respectively.

From our supposition, $\tilde{S}^{*,(\ell+1)}$ is not an optimal solution to Problem 2 for cardinality ℓ (its value is strictly worse than that of $S^{*,(\ell)}$). We next consider constructing a solution of cardinality $\ell+1$ from $S^{*,(\ell)}$ by adding a unit of budget for community $\tilde{i}$. With the nesting of the budget (6) for community $\tilde{i}$ specifically, by Assumption 1 the resulting marginal gain must be at least as large as the marginal gain by adding a unit of budget to community $\tilde{i}$ for $\tilde{S}^{*,(\ell+1)}$. (Recall from Remark 1 that the assumption holds due to submodularity if the subsets chosen by `sol-method` are nested.) Thus,

$$\overbrace{\left[\sum_{i=1}^{c}\sigma_i(S_{i,k_i^{*,(\ell)}})\right]}^{\text{Value of } S^{*,(\ell)}}+\overbrace{\left[\sigma_{\tilde{i}}(S_{\tilde{i},k_{\tilde{i}}^{*,(\ell)}+1})-\sigma_{\tilde{i}}(S_{\tilde{i},k_{\tilde{i}}^{*,(\ell)}})\right]}^{\text{Marginal gain of augmenting } S^{*,(\ell)}}$$

$$\geq \overbrace{\left[\sum_{i=1}^{c}\sigma_i(S_{i,k_i^{*,(\ell)}})\right]}^{\text{Value of } S^{*,(\ell)}}+\overbrace{\left[\sigma_{\tilde{i}}(S_{\tilde{i},k_{\tilde{i}}^{*,(\ell+1)}})-\sigma_{\tilde{i}}(S_{\tilde{i},k_{\tilde{i}}^{*,(\ell+1)}-1})\right]}^{\text{Marginal gain of augmenting } \tilde{S}^{*,(\ell+1)}}, \qquad \text{(by Assumption 1)}$$

$$> \overbrace{\left[\sigma_{\tilde{i}}(S_{\tilde{i},k_{\tilde{i}}^{*,(\ell+1)}-1})+\sum_{\substack{i=1,\ldots,c\\ i\neq\tilde{i}}}\sigma_i(S_{i,k_i^{*,(\ell+1)}})\right]}^{\text{Value of } \tilde{S}^{*,(\ell+1)}}+\overbrace{\left[\sigma_{\tilde{i}}(S_{\tilde{i},k_{\tilde{i}}^{*,(\ell+1)}})-\sigma_{\tilde{i}}(S_{\tilde{i},k_{\tilde{i}}^{*,(\ell+1)}-1})\right]}^{\text{Marginal gain of augmenting } \tilde{S}^{*,(\ell+1)}}, \qquad (\tilde{S}^{*,(\ell+1)} \text{ is not optimal})$$

$$= \overbrace{\left[\sum_{i=1}^{c}\sigma_i(S_{i,k_i^{*,(\ell+1)}})\right]}^{\text{Value of } S^{*,(\ell+1)}}.$$

Thus, the objective value of the optimal budget allocation $\mathbf{k}^{*,(\ell+1)}$ for a budget of $\ell+1$ is strictly less than the objective value of a budget allocation we constructed. This is a contradiction. Thus our assumption about an instance lacking nesting (up to uniqueness) was incorrect. □

APPENDIX E
COMPUTATIONAL COMPLEXITY ANALYSIS

In this section, we analyze the computational complexity of the proposed framework (Algorithm 1). The run-time of the proposed framework is the sum of the times taken at the three steps. It depends on the choice of community detection method as well as the solution method to solve IM for each community. We analyze the run-time involved at each step as follows.

A. Learning the inherent community structure of the social network

The worst-case run-times of different community detection algorithms considered in this paper are given as follows: the Louvain method is $O(n\log n)$ [32], label propagation is $O(n+|E|)$ [33], and the Girvan-Newman method is $O(n|E|^2)$ [46].

B. Generating candidate solutions by solving the influence maximization problem for each community

If we use CELF++ to solve IM for c different communities then we are solving c problems of finding a k-node subset for each community from n_i nodes, $i=1,\ldots,c$. For the ith community, CELF++ iteratively builds the k-node subset as follows. First, find the best individual node by evaluating all n_i subsets of cardinality one. Next, find the node with the highest marginal influence in the presence of the best individual node by evaluating (up to) all n_i-1 subsets of the previously selected best individual and an additional node. CELF++ then keeps adding nodes to the previous set in the same manner until the size of the current set is k. The number of k-node subsets evaluated at the kth step is $n_i-(k-1)$ in the worst case. Thus, the number of subsets evaluated in the worst case is

$$\sum_{i=1}^{c}[n_i+(n_i-1)+\cdots+(n_i-(k-1))]$$
$$=nk-\frac{ck(k-1)}{2}. \tag{7}$$

On the contrary, if we use CELF++ for the entire network then the total number of subsets evaluated in the worst case is

$$n+(n-1)+\cdots+(n-(k-1))=nk-\frac{k(k-1)}{2}. \tag{8}$$

By comparing (7) and (8), we observe that the Generate-Candidates step of the proposed framework achieves a lower run-time compared to using the `sol-method` for the entire network by an additive factor of $(c-1)k(k-1)/2$. Furthermore, as $n_i\leq n\ \forall i=1,\ldots,c$, the length of the diffusion while evaluating a subset of the nodes using Monte Carlo simulations within any community will always be smaller as compared to doing the same in the entire network. This further reduces the run-time of the Generate-Candidates step.

C. Final seed set selection using progressive budgeting

The progressive budgeting method of final seed set selection solves 'finding the maximum of c elements' k times. Hence, the worst-case run-time of progressive budgeting is $O(ck)$.

In practice, solving IM for each community (using a simulation-based `sol-method`) is the step that takes the most amount of time due to the costly Monte Carlo simulations. In that sense, the worst-case run-time of the proposed framework (with a simulation-based `sol-method`) to solve IM for each community is lower compared to the same for solving IM for the original network using the same simulation-based `sol-method`.