IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 10, 2019, accepted March 1, 2019, date of publication April 4, 2019, date of current version April 19, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2906081

InfoFlow: Mining Information Flow Based on User Community in Social Networking Services

JOSUE OBREGON[1], MINSEOK SONG[2], AND JAE-YOON JUNG[1]
[1]Department of Industrial and Management Systems Engineering, Kyung Hee University, Yongin 446-701, South Korea
[2]Department of Industrial and Management Engineering, Pohang University of Science and Technology (POSTECH), Pohang 37673, South Korea

Corresponding author: Jae-Yoon Jung (jyjung@khu.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) through the Korean Government (MSIP) under Grant 2013R1A2A2A03014718.

ABSTRACT Online social networking services (SNSs) have emerged rapidly and have become huge data sources for social network analysis. The spread of the content generated by users is crucial in SNS, but there is only a handful of research works on information diffusion and, more precisely, information diffusion flow. In this paper, we propose a novel method to discover information diffusion processes from SNS data. The method starts preprocessing the SNS data using a user-centric algorithm of community detection based on modularity maximization with the purpose of reducing the complexity of the noisy data. After that, the InfoFlow miner generates information diffusion flow models among the user communities discovered from the data. The algorithm is an extension of a traditional process discovery technique called the Flexible Heuristics miner, but the visualization ability of the generated process model is improved with a new measure called response weight, which effectively captures and represents the interactions among communities. An experiment with Facebook data was conducted, and information flow among user communities was visualized. Additionally, a quality assessment of the models was carried out to demonstrate the effectiveness of the method. The final constructed models allowed us to identify useful information such as how the information flows between communities and information disseminators and receptors within communities.

INDEX TERMS Information flow, social networking services, community detection, network modularity, process mining.

I. INTRODUCTION

Social Networking Services (SNS) have spread globally and generate huge amounts of data every day. Billions of people around the world connect to friends, family, co-workers, schoolmates, and acquaintances through these services. For example, Twitter had 328 million monthly active users in June 2017. Facebook reported an average of 2.01 billion monthly active users as of June 30, 2017. If we add these users to the number of users of other SNS such as Google+, LinkedIn, or Instagram, for instance, the number increases substantially. SNS generally offer users the option to create a profile to share content such as messages, opinions, photos, and videos. Furthermore, SNS allow users to create social relationships by connecting with other users. The definition or nature of this connection varies between different SNS. Taking Facebook and Twitter as examples, a connection on Facebook is called friendship, with both users agreeing to establish the social relationship. In contrast, the connection on Twitter is represented by a following action, in which followed users do not have to approve the relationship, and the followed users do not have to follow their followers. When a user posts or publishes content, other users can interact using comments or spread the content using different mechanisms of the SNS; for example, shares on Facebook and retweets on Twitter. When these actions are repeated continuously, various processes occur among the users, such as information dissemination [1]. As a result, there are huge amounts of data generated every moment from SNS, which may contain hidden user interactions that can be discovered.

Process mining is a methodology that has gained more attention over the last decade. Process mining is built upon data mining principles and deals mainly with business

The associate editor coordinating the review of this manuscript and approving it for publication was Yichuan Jiang.

execution data stored by information systems. One branch of process mining is called process discovery, which mines a process model from a stored event log such as business execution records [2]. Nevertheless, SNS data is at first glance not suitable for process mining. With some preprocessing, the data can be adapted and transformed into a process mining input artifact.

In this paper, we develop a method for discovering information diffusion models from SNS data by applying process mining techniques. We first filter the data by removing infrequent users. Then, a user-centric clustering technique based on *modularity* maximization [7] is performed to reduce the complexity of the SNS filtered data. The user-centric approach was achieved using a novel measure called the *user intimacy* value, which measures the relationship level between users. The clustered data are transformed into an event log and used as an input artifact for a newly developed process discovery algorithm named *InfoFlow miner*, which is based on the Flexible Heuristics miner [8]. The algorithm uses another novel measure called *response weight*, which is defined as the extent of influence or impact that one user's actions has on another user's actions. The final result of the method is a graph representing the information flow between user communities contained in the original data.

The contribution of this research is twofold. First, the complexity of noisy SNS data can be reduced using a user-centric approach by introducing the *user intimacy* measure, so that further analysis using the detected communities such as market segmentation can be performed. Second, an effective visualization method was developed for discovering information diffusion flows directly from SNS data with an extended process mining technique.

The remainder of this paper is structured as follows. In Section II, background related to social network analysis and process mining is introduced. In Section III, the general overview and explanation of the framework that supports the proposed approach is described. Sections IV and V give detailed information on the techniques used in this paper. The implementation using different tools is described in Section VI, which also provides the results of the evaluation. Section VII reviews related works. Finally, Section VIII concludes the paper and gives insight for future work.

II. BACKGROUND

A. SOCIAL NETWORK ANALYSIS

A social network is a structure in which the nodes are people, and the edges represent some kind of relationship or interaction between connected people. Social network studies began in the 1930s with Moreno [4] and have become an interesting topic among researchers. When people hear about social networks, words like Facebook or Twitter come to mind. However, Facebook and Twitter are examples of social networking services, also known as SNS, that provide platforms to create social network data on the Web [5].

Social network analysis uses standard measures and metrics for quantifying network structure to understand and explain behavior and interactions between members [5]. Properties like *centrality*, *page rank*, *closeness*, and *betweenness* (among others) are used to describe the structural characteristics of a network. One measure that is important for this research is *modularity*. To understand *modularity*, we briefly introduce the concepts of homophily and assortative mixing. These concepts are common in social network studies and refer to the fact that people generally have a strong tendency to associate with others whom they perceive as being similar to themselves in some way. Moreover, assortative mixing can be quantified. A network is considered assortative if a significant fraction of the edges in the network run between same types of node. If we find the fraction of edges that run between nodes of the same type and then subtract the fraction of such edges that we would expect to find if edges were positioned at random without considering the type of node, we can quantify the networ's assortative mixing. This quantity is called *modularity* [5]. The standard equation for calculating *modularity* is given by:

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta \left(c_i, c_j \right) \tag{1}$$

where m is the number of edges in the network, A is the adjacency matrix of the network, k is the degree of the node, c is the class or type of node, and $\delta(c_i, c_j)$ is the Kronecker delta function.

The community detection algorithm used in our approach is based on *modularity* maximization, which moves nodes between communities inside the network with the objective of finding the maximum value of *modularity*. The specific algorithm used was introduced by Blondel *et al.* [7].

B. PROCESS MINING

Organizations record historical business process data using information systems that contain event logs. Event logs provide real information about the execution of the business processes operating inside the organization. Process mining techniques take advantage of that fact to extract knowledge from these records. Process mining is a research discipline that combines data mining techniques with business process modeling and analysis to discover, monitor, and improve real processes. There are three areas of process mining: process discovery, conformance verification, and enhancement. Process discovery algorithms generate a process model based only on actual process execution data found in event logs. Conformance verification deals with the analysis of process models and event logs with the objective of finding differences between the process model and the actual execution of the process. Finally, process enhancement extends and improves current process models using information from the event log [2].

The process mining type used in this research is process discovery. A more detailed, but not formal, definition of a process discovery algorithm is a function that maps an event log onto a process model such that the model represents the behavior seen in the event log. There are many process

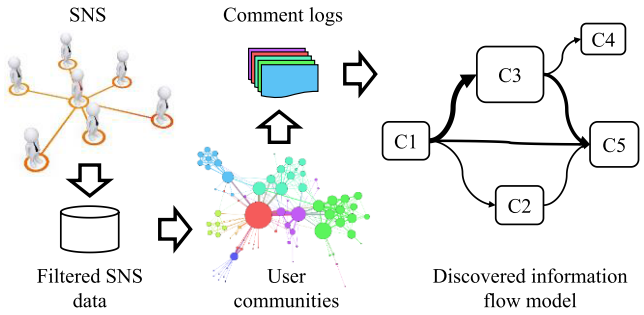

FIGURE 1. Framework for mining information flow of SNS data.

discovery algorithms including alpha miner, heuristics miner, and inductive miner [2], [8]. Every algorithm has its strengths and weaknesses. A multi-dimensional quality assessment of several process discovery algorithms is presented in [9]. The authors evaluated the algorithm using quality measures including accuracy, precision, recall, and comprehensibility. In one of their conclusions, they state that the Heuristics miner [8] works well with real-life context data in terms of accuracy, comprehensibility, and scalability. Therefore, it appears to be appropriate for data with considerable noise. This is a crucial characteristic that is desirable for analyzing SNS data. Therefore, we base our technique on the Heuristics miner algorithm adapted to our purpose. This is explained in detail in Section V. Next, we provide an overview of our framework and a description of the data used as input for our technique.

III. INFORMATION FLOW MINING

In this section, the overall framework of the proposed approach is described. Moreover, the common structure and main characteristics of SNS data are introduced, as are the assumptions that allow the use of process mining techniques for this type of social data.

A. FRAMEWORK

The proposed approach is based on the "divide and conquer" statement used by Song *et al.* [10], which attempts to give a solution to diversity, i.e., the condition of having or being composed of differing elements. In our user-centric technique, this refers to diversity of types of users.

The overview of the proposed methodology for mining information flow is shown in Fig. 1. SNS provide public interfaces for accessing data. Different SNS platforms exhibit similar basic data structures and attributes. Therefore, it is possible to apply our methodology to different SNS. Using the public interface, SNS data are gathered and filtered to reduce noise. The required data attributes are stored systematically for posterior use. A user clustering technique based on *modularity* maximization [7] is applied to the SNS data to obtain communities of users. A comment log file is generated based on the communities found in the previous step. Finally, we apply the *InfoFlow miner*, which is based on the Flexible Heuristics Miner [8], to generate models of information diffusion.

TABLE 1. Comparison between business data and SNS data.

Business data	SNS data
A process consists of business cases	A process consists of posts
A case consists of events related to precisely one case	A post consists of user actions related to precisely one post
Operation events within a case are ordered	User actions within a post are ordered
Operation events can have attributes such as time, cost, and resource	Users' actions can have attributes such as likes, timestamp, and location

B. SOCIAL NETWORKING SERVICE DATA

Because traditional process discovery algorithms use business process data as input, it is important to understand how SNS data is structured and how the algorithm works to define the differences between SNS data and business process data. However, both differences should be considered because similarities also have great importance since SNS data should be adapted to the general structure of the process discovery algorithm input artifacts. Therefore, understanding the similarities can help us establish our assumptions in such a way that we can justify the use of SNS data as an input for process mining.

The challenge is to find a way to relate SNS data with business process data. Despite the differences that exist between these kinds of data, we need to construct an event log that will serve as an input for a process discovery algorithm if we want to use a process discovery technique. Fortunately, the two kinds of data exhibit some similar key characteristics that enable us to generate an event log file from SNS data.

The key characteristics on which we base our assumptions for the construction of event logs of comments are summarized in Table 1. Business process data characteristics were presented by van der Aalst in [2]. A business process consists of a set of cases. Cases consist of events related precisely to one case. Events within a case are time ordered. Finally, events can have attributes including time, cost, or resources that can be used to gain extra knowledge from event logs.

For SNS data, a process consists of a set of posts, which are created by users. Each post also contains user actions, such as likes and comments on Facebook and conversations and retweets on Twitter, which are precisely related to one post. User actions within a post are ordered by time. Actions can appear ordered by importance, but the timestamp of each action is recorded. Finally, user actions can have different attributes such as timestamp, likes, and shares on Facebook.

Considering the key similarities presented, a *comment log* from SNS data representing user actions can be constructed as an input artifact for a process discovery algorithm. This will be defined in Section V, but first we explain the user clustering technique for discovering communities.

IV. USER COMMUNITY DETECTION

In this section, we introduce the important definitions in the proposed approach. Moreover, we introduce the community detection algorithm that was utilized to reduce the complexity of SNS data by grouping similar users in communities based on their relationships.

In this paper, a community is defined as a group of users who tend to have frequent actions inside one or more posts. Communities are discovered based only on information found in the data log stored in the SNS. In other words, we do not use any prior knowledge about the social relationships between users. The interactions between users are measured with an intimacy function that quantifies how related any two users are based on the frequencies of their common actions in SNS. User communities are detected based on the intimacy values obtained from the data.

A. USER INTIMACY

First, there is a set of users who interact in a SNS, denoted by $U = \{u_k | \, k = 1, \ldots, K\}$, where K is the number of users in a dataset. For a given set U, U^* is the set of all finite sequences over U. U^* can then represent any sequence of users in U.

SNS generally contain posts written by users. Each post is accompanied by a set of comments for the post. We define a set of posts in the SNS as $P = \{p_n | \, n = 1, \ldots, N\}$, where N is the number of posts in the dataset. For a specific post, the comment sequence of users who commented is defined as follows.

Definition 1 (Comment Sequence): For a post p_n, the comment sequence of users who commented in the post, denoted by σ_n, is defined as $\sigma_n =< u_1, u_2, \ldots, u_M >$, where $u_m \in U$ is the user who has written the m-th comment for p_n, and M is the number of comments associated with p_n, i.e., $M = |\sigma_n|$. Hence, a comment log in SNS data can be represented as a multiset of comment sequences over all posts, denoted by $L = [\sigma_n]$ with $n = 1 \ldots N$.

We note that σ_n is a subset of U^* because σ_n is a sequence of users in U, i.e., $\sigma_n \in U^*$. Herein, a user may appear more than one time in σ_n.

To measure *user intimacy*, we assume that, if two users frequently comment in the same post, their intimacy value increases. Under this assumption, the intimacy of two users can be defined for comment sequences of users in L.

Definition 2 (User Intimacy): For a comment log L, the user intimacy between two users i and j, denoted by τ_{ij}, is defined as the frequency at which user i has preceded user j in all comment sequences in L.

$$\tau_{ij} = \sum\nolimits_{\sigma_n \in L} |u_i \succ u_j|_{\sigma_n} \tag{2}$$

where $|u_i \succ u_j|_{\sigma_n}$ is the frequency at which u_i has preceded u_j in a comment sequence σ_n. Moreover, a user intimacy matrix $T = (\tau_{ij})$ is a matrix containing user intimacy values between all users $u \in U$ who appear in L.

We present an example of a *user intimacy matrix* calculation in Fig. 2. There are three comment sequences (σ_1, σ_2, and σ_3) with three users interacting (u_1, u_2, and u_3). F, the *user intimacy* value between users u_1 and u_2, is calculated. Using σ_1, $|u_1 \succ u_2|_{\sigma_1} = 3$ because u_2 is preceded by u_1 three times in σ_1. From σ_2, we have $|u_1 \succ u_2|_{\sigma_2} = 1$ because u_2 is preceded by u_1 only once in σ_2. Likewise, the value for σ_3 is calculated as $|u_1 \succ u_2|_{\sigma_3} = 2$. The final *user intimacy* value between users u_1 and u_2 is equal to the sum of all the values calculated before, i.e., $\sum_{\sigma_n \in L} |u_1 \succ u_2|_{\sigma_n} = 3 + 1 + 2 = 6$. The other *user intimacy* values are calculated in the same way, resulting in a K squared matrix, where K is the number of users in the dataset.

Comment sequences

$\sigma_1 =< u_1, u_2, u_3, u_1, u_3, u_2 >$

$\sigma_2 =< u_3, u_2, u_1, u_3, u_3, u_2 >$

$\sigma_3 =< u_1, u_1, u_3, u_2 >$

User intimacy matrix T

	u_1	u_2	u_3
u_1	2	6	7
u_2	2	2	4
u_3	2	7	4

FIGURE 2. Example of a *user intimacy matrix.*

Note that the *user intimacy matrix* is not symmetric. The *user intimacy*values between users u_1 and u_2 are not the same compared with *user intimacy*values in the opposite direction, that is, the value calculated starting from user u_2 to user u_1.

B. MODULARITY MAXIMIZATION

A directed weighted network $G^0 = (V^0, E^0)$ is created based on *user intimacy matrix* T. The nodes of the network, V^0, are the set of users U. The weight and direction of edges between nodes, E^0, are extracted from each element τ_{ij} of the matrix T.

The community detection algorithm used in the proposed approach is a heuristic method introduced in [7] based on *modularity* maximization. The concept of *modularity* was introduced in Section II and is a measure that quantifies the network's tendency to be partitioned into modules or groups.

Algorithm 1 describes the community detection procedure. The algorithm is divided into two phases that are repeated iteratively. In the first phase, a different community is assigned to each network node. A node i and its neighbors are considered for each network node. The community of i is interchanged with each community of its neighbors, and the *modularity* is evaluated in each case. Node i is then placed in the community where the *modularity* gain is maximized. This is repeated until no further improvement can be achieved.

In the second phase, a new network is built whose nodes are now the communities found during the first phase. The weights of the links between the new nodes are given by the sum of the weights of the links between nodes in the corresponding two communities [7]. Blondel *et al.* [7] performed computer simulations on large ad-hoc modular networks and the complexity of their community detection algorithm is linear.

In this procedure, the so-called *resolution* parameter can be used. Community detection techniques based on *modularity* maximization are affected by a parameter called *resolution*

Algorithm 1 Community Detection

Input: Weighted and directed network $G^0 = (V^0, E^0)$
Output: Partitioned network by communities $G^m = (V^m, E^m)$

```
Initialize
 Assign a different community to each node of G^0
Repeat
 For each node i
  For each neighbor j from i
   Interchange communities between node i and
   node j looking for modularity gain
  End for
 End for
 Construct a new network G^m using the communities
 found on the previous phase as nodes
Until a maximum in the modularity is reached
Return G^m = (V^m, E^m)
```

that allows exploration of the cluster structure of a graph at different scales [11]. Varying the *resolution* parameter can affect the number of communities obtained from a network. Lower resolution values (generally less than 1.0) generate more communities, while higher resolution values (generally greater than 1.0) generate fewer communities.

The final step of user community detection is to generate a comment log among user communities. In our work, we focus on the community instead of the user itself to capture information flow in SNS. Hence, each user in the comment log, described in Definition 1, is substituted with the community in which the user belongs based on the result of the community detection algorithm. The resulting comment log L is used as the input for process discovery in the next step of the method.

V. PROCESS DISCOVERY

In this section, we present the *InfoFlow miner* used for discovering process models from SNS data. Herein, an *information flow* is a directed graph that represents the process by which information propagates among users or groups of users in SNS. We then explain four special cases of assumptions for measuring interactions among users inside a comment log L.

A. RESPONSE WEIGHT

The process discovery algorithm presented in [8] was extended for this work. The Flexible Heuristics miner algorithm generates process models even in the case of non-trivial constructs, low structure domains, and the presence of noise. However, some modifications to the algorithm calculations must be made to obtain the desired information diffusion process results.

The Flexible Heuristics miner algorithm considers frequencies of events and sequences when constructing the process model. The goal is that infrequent patterns should not appear in the generated process model [2], [8]. To accomplish this, the "*direct follow*" frequency between activities on the event log measures the number of times one activity is directly followed by another activity. However, we need to relax this measure to allow indirect follows among users since disconnected actions among users or communities (i.e., indirect follows) also have significance for SNS information flow. Therefore, we extend the concept of direct follows to *response weight*.

Definition 3 (Response Weight): For a comment log L, the response weight between two users k and l, denoted by w_{kl}, is defined as the amount that user k is affected by user l in all posts.

$$w_{kl} = \sum\nolimits_{\sigma_n \in S} w^n_{kl} \tag{3}$$

where w^n_{kl} represents the response weight in a comment sequence σ_n in S and is calculated as

$$w^n_{kl} = \begin{cases} (Pos_n(k) - 1)^{-\alpha}(pos_n(k) - pos_n(l))^{-\beta} \\ \quad if\ k, l \in \sigma_n\ and\ pos_n(k) > pos_n(l) \\ 0, otherwise \end{cases} \tag{4}$$

where $pos_n(k)$ is the position of user k in σ_n, and α and β are parameters that allow adjustments to the response weight.

Response weight measures the influence or impact of one user's actions on another user's actions. It also denotes the response reflection of a user. As shown in Fig. 3a, the part of the equation modified by the parameter α (i.e., $pos_n(k) - 1$) represents the total number of preceding users in relation to user k. In contrast, the part of the equation modified by the parameter β (i.e., $pos_n(k) - pos_n(l)$) represents the distance between user k and its preceding user l. Varying parameters α and β allows us to study different weighting definitions. Some special variations are depicted in Fig. 3b, 3c, 3d, and 3e and are detailed in the next section.

B. SPECIAL CASES OF RESPONSE WEIGHT

Four special cases for calculating response weights between users using different values of α and β parameters are introduced below:

1. Evenly distributed weighting (EDW): $\alpha = 1$ and $\beta = 0$, $w^n_{kl} = (pos_n(k) - 1)^{-1}$ for every outgoing edge from node k.
2. Uniform weighting (UW): $\alpha = 0$ and $\beta = 0$, $\mathrm{w}^{\mathrm{n}}_{\mathrm{kl}} = 1$ for every outgoing edge from node k.
3. Inverse distance weighting (IDW): $\alpha = 0$ and $\beta = 1$, $w^n_{kl} = (pos_n(k) - pos_n(l))^{-1}$ for every outgoing edge from node k to node l.
4. Adjacency weighting (AW): $\alpha = 0$ and $\beta = \infty$, $w^n_{kl} = (pos_n(k) - pos_n(l))^{-\infty}$ for the single outgoing edge from node k to l when $l = k - 1$, otherwise $w^n_{kl} = 0$.

The four special cases are demonstrated graphically in Fig. 3(b). EDW states that the weight of the response of user k is evenly distributed to all its preceding users. Consequently, the weight of the answer of the latter users is very small compared with the weight of the answer of the

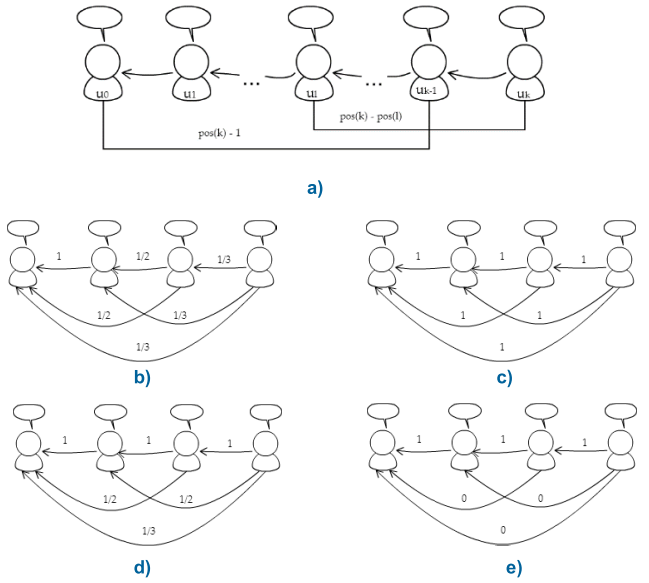

FIGURE 3. (a) Response weight diagram. The important measures are the distance between user *k* and user *l* and the distance between user *k* and the first user of the comment log *L*. (b) EDW, the response influence value of a user, is evenly distributed among all its preceding users. (c) UW, a response influence value, is uniform between all users. (d) IDW, a response influence value, is the inverse of the distance between a user and each of its preceding users. (e) AW, a response influence value, only has a value between adjacent users.

first users. This weight gives more emphasis to users who comment first in a sequence of comments. These users have more weighted connections than other users, reflecting more interaction in the generated model. (c) UW states that the weight of the response of user k is equal to 1 for all preceding users no matter the distance between them. This implies that a user's answer uniformly influences all its preceding users. Because the weight is uniform, models generated with this *response weight* measure take into account only the frequency of user comments, ignoring the distance between appearances. (d) IDW states that the weight of the response of user l to its preceding user k is equal to the inverse distance between the two users. In this case, the user's response has more influence over its closest preceding users. Models generated in this way take into account the frequency of appearances of the comments and the distance between appearances. (e) AW states that the weight of the response of user l to its preceding user k is equal to 1 only if they are adjacent on the sequence of comments, which means that the user's response only influences the immediately preceding user. Models generated using this *response weight* measure have weighted connections only between users that commented consecutively.

C. INFORMATION FLOW GRAPH

An Information Flow Graph (IFG) can be constructed, which is similar to the Dependency Graph (DG) used in the original heuristics miner [8].

Definition 4 (Information Flow Graph): The information flow graph is a tuple IFG = (CA, IF),where CA is the set of communication agents through whom the information flows, and IF ⊆ CA × CA is the set of edges that connect two communication agents and represents information flow between edges.

The communities obtained using the community detection technique are represented as communication agents in the IFG and each edge $if \in IF$ has a direction and weight. In our case, this weight is represented by the *response weight* measure.

Algorithm 2 InfoFlow Miner

Input: Comment log L
Output: Weighted Graph $G^f = (V^f, E^f)$
1: Assign the cluster of user names to V^f
2: Create an empty response weight matrix $\boldsymbol{M}$
3: **for** every comment sequence σ_n in L **do**
4: **for** every comment a_k in σ_n **do**
5: **for** every antecedent comment a_l in σ_n **do**
6: calculate response weight w_{kl}
7: update M with w_{kl} for comments a_k and a_l
8: **end for**
9: **end for**
10: **end for**
11: Assign response weights in $\boldsymbol{M}$ to E^f
12: **return** $G^f = (V^f, E^f)$

The *InfoFlow miner* general procedure is described in Algorithm 2. The first step is the creation of a K-square matrix containing the row headers and column headers equal to all the users (or group of users) $u_m \in U$. After that, the comment log L is traversed entirely. For each $u_m \in \sigma_n$, the response weight from user u_m to each of the remaining M users until the end of the sequence is calculated and updated in the response weight matrix. Finally, the response weight matrix is transformed into an $IFG = (CA, IF)$, where $CA = V^f$ and $IF = E^f$. *InfoFlow miner* is driven by the ordering relations between user comments. *InfoFlow miner* is driven by the ordering relations between user comments. The time complexity used to build these relations is linear, and the complexity of the remaining steps in the algorithm are exponential in the number of tasks [12]. Nevertheless, in our case the tasks are the communities, and usually the number of communities is less than 100. Therefore, it can be said that the computation time of InfoFlow is not serious because of the small number of communities.

VI. EXPERIMENTS

In this section, we describe the steps taken for implementing the proposed method. *InfoFlow miner* was developed on ProM, which is a well-known framework for process mining[1]. An experiment using Facebook data was performed and is used as a running example to show the implementation of each stage.

[1] http://www.promtools.org/

TABLE 2. Results of the community detection algorithm.

Resolution	Modularity	Communities
0.5	0.079	24
0.6	0.106	18
0.7	0.137	14
0.8	0.171	14
0.9	0.203	12
1.0	0.241	9

A. DATA

The first task performed was collection of SNS data from Facebook. For this purpose, we used the Facebook Graph API, which is the primary way to retrieve or post data from Facebook. Additionally, an XES file was generated with SNS data using the structure defined in Section III. We gathered data from the Facebook CNN Fan Page. The data consisted of all posts made by CNN from May 17th to May 31st, 2013. The data is comprised of 298 posts with 38,374 users generating 79,507 comments. During the second stage of our approach, the data was filtered to reduce noise (i.e., infrequent users). The filter retrieved the top 20% most frequent commenters, which were users that appeared at least 12 times in the 298 posts. After data filtering, users were reduced to 685 generating 15,904 comments among 292 posts.

B. COMMUNITY DETECTION

The second task of our approach is user clustering, also known in our approach as community detection. The *InfoFlow miner* plug-in was used for this purpose. Panel values of α and β parameters can be set in the plug-in configuration. We can also set the number of times we want to execute the community detection algorithm.

In our experiment, the community detection function (based on *modularity* maximization) was executed six times with six different *resolution* parameter values. The results are shown in Table 2. We started with a *resolution* value of 0.5 and increased it until we reached 1.0. The best results were obtained with a *resolution* value of 0.6. The explanation of this value is given in subsection D. Note that increasing the value of the *resolution* parameter decreases the number of communities discovered and increases the *modularity* value.

Figure 4 shows a graphical representation of the communities of users using a *resolution* parameter value of 1.0. The graph was generated in the Gephi tool using the OpenOrd layout algorithm. Finally, a comment log was generated as an input for the process discovery phase for each repetition of the experiment.

C. PROCESS DISCOVERY

The last step is perhaps the most important part of the approach because we are able to visualize the discovered information diffusion models generated using the *InfoFlow miner*. Figure 5 shows a heuristic model obtained using the Flexible Heuristics miner with the raw Facebook data.

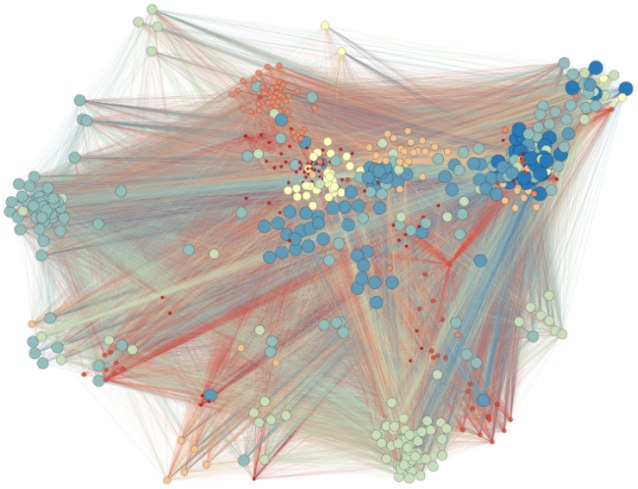

FIGURE 4. Partitioned graph obtained from the community detection algorithm for Facebook data with a *resolution* parameter value of 1.0. The node color denotes the community to which the node belongs.

The model is very complex and unreadable. Therefore, our modeling approach must provide insight into the information diffusion processes that occur within SNS data.

Several process models were generated for the Facebook data. Four rounds of process discovery were performed on each of the six different logs obtained from the previous part of the experiment. The four rounds correspond to the special cases of *response weight* presented in Section V.B. Figure 6 shows the final results of the process discovery from SNS data for the experiments using a *resolution* parameter value of 0.6, which gives the best quality results, as described in the next section.

The nodes generated in the IFGs represent communities of users. The node size is defined according to the number of users belonging to each community. The greater the node size, the more users the community has compared with other communities in the same model. The number inside each node represents the number of user interactions inside the community with other users of the same or different community. Furthermore, the thickness of the edges between nodes is adjusted according to the *response weight* measure between user communities. A thicker edge (e.g., edge between C3 and C0 in Fig. 6 (b)) represents a higher value of response weight compared with the other *response weights* in the same model. Let us now introduce the quality evaluation framework as well as the results for the discovered models from Facebook data.

D. QUALITY EVALUATION

To evaluate the importance and validity of the proposed approach, the quality of the discovered models was measured. The accuracy framework used was presented in [9]. We used a combination of recall and precision measures in the evaluation of the discovered models, as well as two versions of

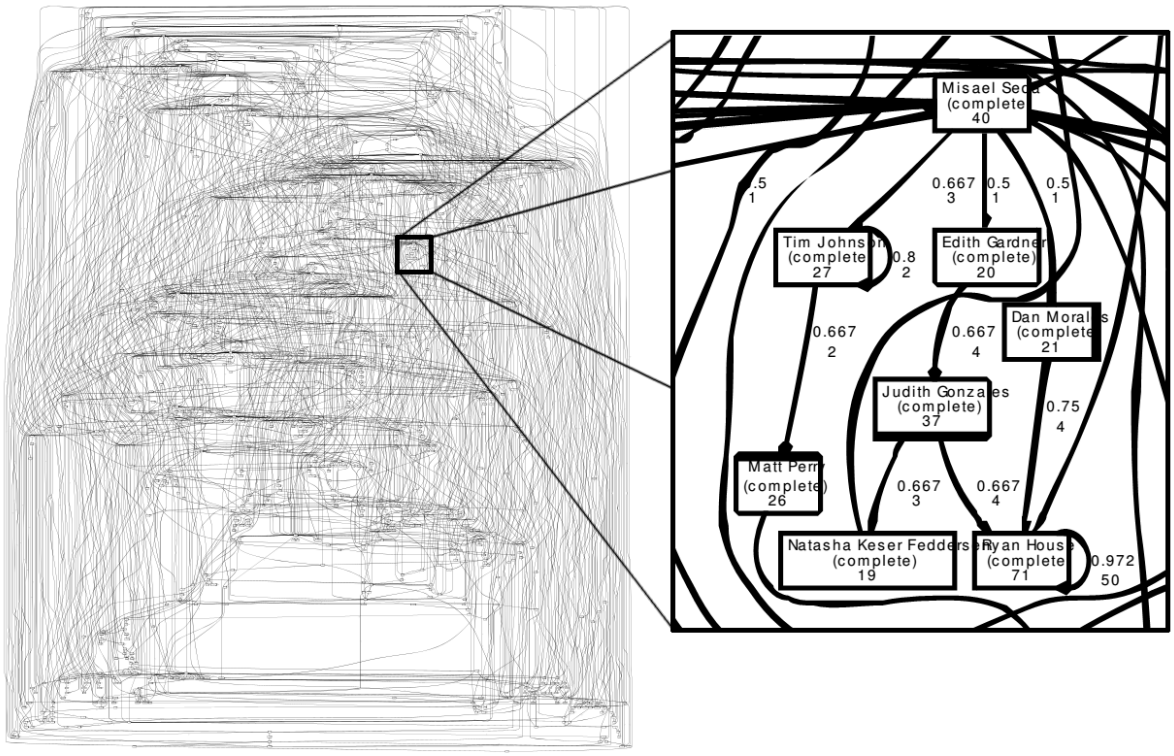

FIGURE 5. Discovered heuristic model from raw Facebook data. The model is complex and unreadable.

F-score because they can be employed to combine different types of accuracy metrics.

First, the recall ability of the process model is measured using the notion of fitness. A model with good fitness allows for the behavior seen in the event log. For this purpose, the heuristic net generated from the *InfoFlow miner* is transformed into a petri net N [2]. After that the event log is replayed using the model to record all the situations where a transition is forced to fire without being enabled, counting the missing tokens as well as the remaining tokens. This technique is called token replay and is performed at the level of events rather than full traces. The fitness of an event log L is given by:

$$f(L,N) = \frac{1}{2}\left(1 - \frac{\sum_{\sigma\in L} L(\sigma)\times m_{N,\sigma}}{\sum_{\sigma\in L} L(\sigma)\times c_{N,\sigma}}\right) + \frac{1}{2}\left(1 - \frac{\sum_{\sigma\in L} L(\sigma)\times r_{N,\sigma}}{\sum_{\sigma\in L} L(\sigma)\times p_{N,\sigma}}\right) \quad (5)$$

where $L(\sigma)$ is the frequency of trace σ in L, and $p_{N,\sigma}$, $c_{N,\sigma}$, $m_{N,\sigma}$, and $r_{N,\sigma}$ are the numbers of produced, consumed, missing, and remaining tokens when replying σ on N, respectively. The value of $f(L,N)$ is between 0 and 1, where values close to 0 indicate poor fitness and a value of 1 indicates perfect fitness [2].

The notion of recall for the models generated by *InfoFlow miner* at the level of events is that modeling single interactions between communities have more importance than modeling the total sequence of information diffusion from the beginning until the end. The discovered process models are useful because they depict the most representative information diffusion interactions among communities from the comment log. Additionally, we can say that if a model has 0.92 of recall for example, the model is able to explain around 92% of the information diffusion interactions observed in the comment log.

Second, the precision of the process model is evaluated using the *simple behavioral appropriateness* (sa_B) measure. This metric measures the degree of accuracy in which the model describes the observed behavior in the event log by determining the mean number of enabled transitions during log replay [13].

The *simple behavioral appropriateness* measure is given by:

$$sa_B = \frac{\sum_{\sigma\in L} L(\sigma)\,(|T_V| - x_\sigma)}{(|T_V|-1)\times\sum_{\sigma\in L} L(\sigma)} \quad (6)$$

where x_σ is the mean number of enabled transitions in σ during log replay, and T_V is the set of visible transactions in the petri net N.

The notion of precision on the discovered information diffusion models comes from the fact that measuring sa_B is formulated with the notion of the control flow perspective of the process. This means that sa_B measures the degree of accuracy and clarity in which the cascades of comments in an information diffusion process are described and represented by the model.

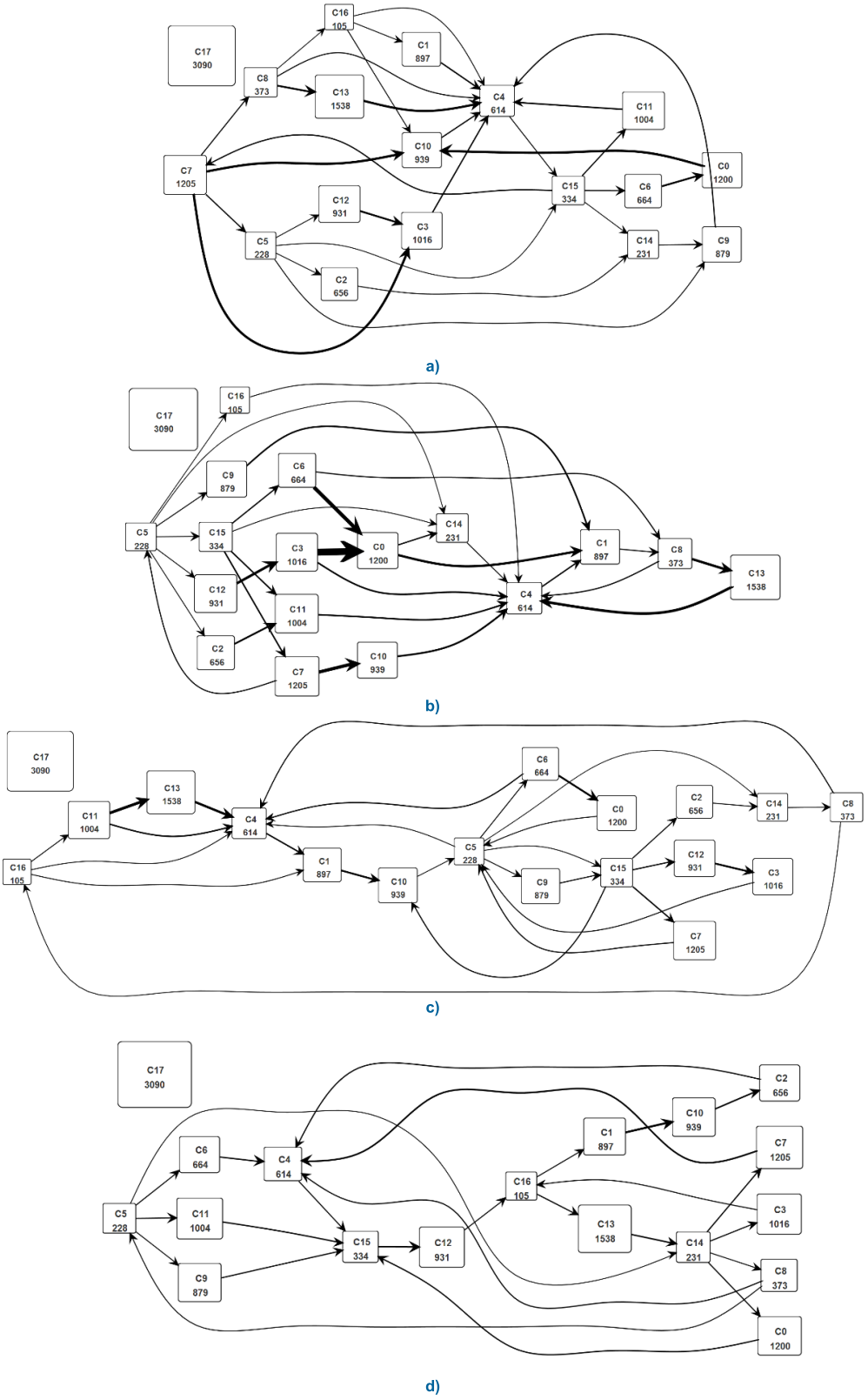

FIGURE 6. Discovered process models using the proposed methodology for Facebook data for the four *response weight* special cases (*resolution* = 0.6). The sizes of the nodes are given by the number of users belonging to the respective community. The number inside the nodes represents the number of interactions of the users inside that community. The thickness of the arcs is determined by the *response weight*. a) Evenly distributed weight (EDW), b) uniform weight (UW), c) inverse distance weight (IDW), d) adjacency weight (AW).

TABLE 3. Quality results obtained from the Facebook experiment.

Resolution	Weight	f	sa_B	F_1	F_2
0.5	EDW	0.8643	0.5383	0.6634	0.7709
	UW	0.8662	0.5163	0.6469	0.7628
	IDW	0.8706	0.5373	0.6645	0.7745
	AW	0.8719	0.5491	**0.6738**	**0.7802**
0.6	EDW	0.8969	0.5429	**0.6763**	**0.7934**
	UW	0.8965	0.5376	0.6721	0.7909
	IDW	0.9002	0.5328	0.6694	0.7911
	AW	0.9007	0.5283	0.6660	0.7894
0.7	EDW	0.9174	0.4967	**0.6445**	**0.7845**
	UW	0.9182	0.4850	0.6347	0.7790
	IDW	0.9097	0.4463	0.5988	0.7533
	AW	0.9128	0.4679	0.6187	0.7670
0.8	EDW	0.8886	0.4288	0.5784	0.7317
	UW	0.8977	0.4592	**0.6075**	**0.7537**
	IDW	0.9017	0.4224	0.5753	0.7349
	AW	0.9042	0.4333	0.5859	0.7428
0.9	EDW	0.9171	0.4707	0.6221	0.7709
	UW	0.9167	0.4760	**0.6266**	**0.7735**
	IDW	0.9144	0.4016	0.558	0.7283
	AW	0.9051	0.4104	0.5647	0.7293
1.0	EDW	0.9282	0.3922	**0.5514**	**0.7289**
	UW	0.9219	0.3535	0.5110	0.6976
	IDW	0.9319	0.3273	0.4845	0.6805
	AW	0.9298	0.3199	0.4760	0.6731

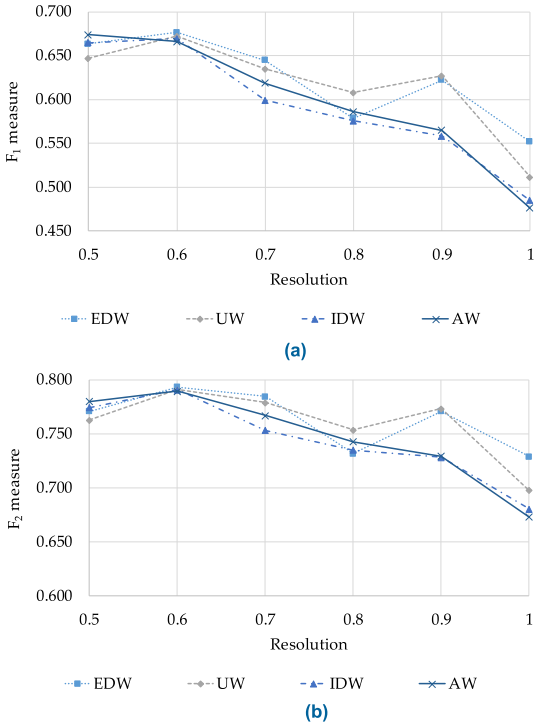

FIGURE 7. Quality evaluation results. The best results were achieved with a *resolution* parameter value of 0.6 for both (a) F_1 and (b) F_2 measures.

A balance between fitness and precision should exist for the discovered models to be considered representative of a process occurring within SNS data. For that reason, we use the same F-measure proposed in [9] for both recall and precision metrics:

$$F_\beta = \left(1 + \beta^2\right) \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall} \tag{7}$$

The parameter β is a weight factor affecting the importance of precision and recall. Two values of the F_β measure are presented. F_1 gives equal weight to precision and recall, while F_2 gives twice as much weight to recall as precision. For information flow models representing SNS data, it is very important that the model captures the community interactions seen in the comment log. In our work, we want to describe how information flows between groups of users. Therefore, the model's ability to describe and replay the interaction between communities, measured by recall, is more important than the sensitivity of the discovered model, measured by precision. In other words, the quality evaluation does not place emphasis on capturing the restrictions of the information diffusion process.

The quality results obtained from all the models generated are shown in Table 3, which summarizes the *resolution* parameter values, *response weight* approach, fitness f measure, sa_B measure, F_1 measure, and F_2 measure for all the process models. As shown in Table 3, there is an interesting relationship between recall and precision with the *resolution* parameter in the experiments. The models generated with a *resolution*parameter value of 0.5 have higher precision compared with the models generated with a *resolution* parameter value of 1. Conversely, the models generated with a *resolution* parameter value of 0.5 have lower values of recall compared with the models generated with a *resolution* parameter value of 1. This demonstrates that more detailed models, which are composed of more communities and fewer users per community are able to represent with more accuracy and clarity the information flow contained in the comment logs. In contrast, coarser grained models which are composed of fewer communities and more users per community are not able to represent accurately the information diffusion process. The recall ability of the models is very similar for all the experiments, which means that independently of the *resolution* parameter value, the models are able to describe the most representative community interactions contained in the comment log. The behavior of the F_1 and F_2 measures are depicted in Fig. 7. The values of F_1 and F_2 measures decrease as the *resolution* parameter increases in most cases. The highest quality results are obtained in models generated with a *resolution* parameter value of 0.6. At this *resolution* scale level, the information flow process can be best described because of the model's balance between communities and users per community. The model is able to capture the information propagation process in such a way that it does not lose important aspects of the diffusion process while it restricts enough behavior to satisfy the modeling requirements.

To further prove the effectiveness of the proposed method, we performed a comparison with the state-of-the-art process discovery methods. First of all, we selected the Alpha miner as a baseline [12] since it was the first process discovery algorithm presented in the literature. We also selected two of the most well-known algorithms on process discovery to

TABLE 4. Performance comparison from the Facebook experiment.

Algorithm	f	sa_B	F_1	F_2
Alpha miner	0.0362	0.0090	0.0144	0.0226
Heuristics miner	0.8450	**0.5937**	**0.6974**	0.7790
Inductive miner	**0.9229**	0.0081	0.0161	0.0391
InfoFlow miner*	0.8969	0.5429	0.6764	**0.7934**

*The InfoFlow model was tested under EDW and 0.6 of resolution.

perform the comparison, the Flexible Heuristics miner [8] and the Inductive miner [2]. The comparison was made against the best model generated by *InfoFlow* miner, using evenly distributed *response weight* and 0.6 of *resolution*. The results are presented in Table 4. The baseline algorithm could not handle SNS data at all, showing a very poor performance in all measures. Inductive miner had a very good recall ability, but the model generated is too general and allowed too much behavior having a very low precision. *InfoFlow* miner performed well in general, having the best performance for the F_2 measure which is more useful for evaluating the quality of information flow models from SNS data. For F_1 measure, the best performance was achieved by Flexible Heuristics miner, which in general demonstrated very similar performance compared to *InfoFlow* miner because both algorithms work in a similar way. However, the Heuristics miner and the proposed *InfoFlow* miner have little performance difference in terms of F_1 and F_2. It is because the two algorithms work in a similar way by considering 'follow' relations between two activities, as described in Section V.*A*. Nevertheless, note that the InfoFlow models can be interpreted according to the selection of response weights, while the Heuristics model can consider only the 'direct follow' relations between communities.

E. FINDINGS AND DISCUSSION

In this section, we present a brief discussion of the behavior exhibited by process models discovered using the proposed methodology.

The models generated by the proposed approach are shown in Fig. 6. They belong to the four different response weights and were generated using a resolution parameter of 0.6. Useful information on the behavior of the user communities can be obtained from these models. First, we have the island nodes, such as community C17 in all models in Fig. 6, which are characterized by the absence of information flow edges with other communities. This means that users from these communities mainly share information between themselves. Second, we have the disseminator nodes [3], also known as hubs. Disseminator nodes include community C5 in all models in Fig. 6, community C14 in Fig. 6d, and community C15 in Fig. 6a and 6b, which receive information from few communities but then spread that information to a greater number of other nodes. Finally, we have the receptor nodes [3], also known as sinks. Receptor nodes include community C4 in all models in Fig. 6, which receive connections from many other nodes but either disseminate the information to a significantly reduced number of communities or do not disseminate it at all. The models generated with *InfoFlow miner* aid in the role identification of the communities participating in the information diffusion process contained in the comment log.

VII. RELATED WORK

The first approach to use social network analysis combined with process mining was presented by van der Aalst and Song [14]. However, the metrics defined in Son's work, such as the handover of work metrics or working together metrics, are specialized for enterprise information system event logs and were not applicable for information diffusion in social networks. Additionally, SNS data cannot be used directly because the results are too complex and difficult to interpret. Therefore, in our work, we use the approach of ''divide and conquer'' to reduce the complexity of the data analyzed. This method is not new among process mining researchers. In [15], Bose *et al.* presented a two-phase approach. In the first phase, they extracted common execution patterns from an event log and transformed the log by replacing the activities with their corresponding abstractions. In the second phase, they used a Fuzzy Miner and found simpler, easier to understand process models. However, we sought to group users based on a measure of their relationships using the *user intimacy* value rather than finding common sequences of user actions within the SNS data.

Several studies have been performed that focus on event log clustering in the area of process mining [10], [16]–[21], [22]. In [10], Song *et al.* propose a method for trace clustering in process mining. In their approach, similarities of traces are measured with trace profiles, in which each of them addresses a specific perspective of the log. Trace profiles are based on typical information found in event logs. For example, they propose a transition profile, which uses direct following relations of the trace, or an originator profile, which uses the number of times a user performs an activity. We instead created the *user intimacy matrix T* to measure the strength of the relationship between users to cluster the user into communities based on *modularity* maximization.

There are some works related to SNS data and information diffusion using process mining [23], [24], [25]. In [23] and [24], Kim *et al.* proposed methods for discovering information diffusion processes that reflect actual interactions among users in social networks using process mining techniques. The problem with [23] is that they used information from social networks directly without any preprocessing, making the discovered model very large and sometimes unreadable. In [24], Kim *et al.* tackled the preprocessing problem using the clustering technique presented in [10] to group users in clusters of similar traces. However, *InfoFlow miner* reduces the complexity of the discovered models by clustering users based on the value of their relationship rather than in the comment traces. The hidden Markov model for information diffusion HMMID was introduced in [25]. This model combines hidden Markov models with process mining techniques to identify information flows between users in

a social network. Even though the proposed approach does not use probabilistic formulations, the models presented are useful for identifying roles in the information diffusion process such as information receptors and disseminators [3].

Finally, several works on information diffusion modeling over social networks have been presented. Some of the works are focused on develop dynamic predictive models using different methods such as considering the nodes of the network as intelligent agents who make strategic decisions [26]. Another work goes further by creating an evolutionary game theoretic framework taking into account user interactions [27]. Lastly [28] uses hydrodynamics to predict and describe the spreading process of the information on both temporal and spatial perspectives. Compared to the proposed approach, we do not deal with temporal dynamics and at this stage a mathematical model for information diffusion is not used. Instead, *InfoFlow miner* is a rather simplistic approach that uses process mining to discover the information flow models. In the approaches presented in [29]–[31], the authors studied the role of tie strength in the diffusion of information in social networks and found that it is indeed very important. This concept is similar to the *user intimacy* value that we present in this paper.

VIII. CONCLUSIONS AND FUTURE WORK

A procedure to discover information diffusion models from SNS data is presented in this work. The data can be used to obtain insight about the information diffusion process among users. Our approach is divided into three main steps. First, data is collected and filtered to identify the most active users. Then, a community detection technique based on *modularity* maximization allowed us to group related users into clusters or communities using the *user intimacy* value, which measures the relationship level between users. Finally, we generated process models using the *InfoFlow miner*. Four different ways to discover models were proposed and analyzed based on the *response weight* between users. The approach was implemented using ProM, and several experiments were performed using real world Facebook data. A quality assessment for the discovered models was performed and discussed. The discovered models enabled us to identify important elements from the information diffusion process such as information disseminators and receptors.

In the future, we will conduct additional studies that reflect the semantics of the content spread over social networks using techniques such as text mining and sentiment analysis. Furthermore, user semantics such as locations, languages, and countries of users could be included for a deeper study of the information diffusion process. Finally, we can also investigate temporal dynamics and the network structure of the information diffusion process as a more complex approach.

REFERENCES

[1] A. Guille, H. Hacid, C. Favre, and D. A. Zighed, "Information diffusion in online social networks: A survey," *ACM SIGMOD Rec.*, vol. 42, no. 2, pp. 17–28, 2013.

[2] W. M. P. van der Aalst, *Process Mining: Data Science in Action*. Heidelberg, Germany: Springer, 2016.

[3] M. De Choudhury, "Discovery of information disseminators and receptors on online social media," in *Proc. 21st ACM Conf. Hypertext Hypermedia*, Toronto, ON, Canada, 2010, pp. 279–280.

[4] J. L. Moreno, *Who Shall Survive?: A New Approach to the Problem of Human Interrelations*. Washington, DC, USA: American Sociological Association, 1934.

[5] M. E. J. Newman, *Networks: An Introduction*. New York, NY, USA: Oxford Univ. Press, 2010.

[6] M. Bastian, S. Heymann, and M. Jacomy, "Gephi: An open source software for exploring and manipulating networks," in *Proc. Int. Conf. Weblogs Social Media*, San Jose, CA, USA, 2009, pp. 361–362. Mech.

[7] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, 2008, Art. no. P10008.

[8] A. J. M. M. Weijters and J. T. S. Ribeiro, "Flexible heuristics miner (FHM)," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Paris, France, Apr. 2011, pp. 310–317.

[9] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens, "A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs," *Inf. Syst.*, vol. 37, no. 7, pp. 654–676, 2012.

[10] M.-S. Song, M. Günther, W. M. P. van der Aalst, and J.-Y. Jung, "Improving process mining with trace clustering," *J. Korean Inst. Ind. Eng.*, vol. 34, no. 4, pp. 460–469, 2008.

[11] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, nos. 3–5, pp. 75–174, 2010.

[12] W. M. P. van der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1128–1142, Sep. 2004.

[13] A. Rozinat and W. M. P. van der Aalst, "Conformance testing: Measuring the fit and appropriateness of event logs and process models.," in *Business Process Management Workshops*. Berlin, Germany: Springer, 2006, pp. 163–176.

[14] W. M. P. Aalst, van der and M. Song, "Mining social networks: Uncovering interaction patterns in business processes," in *Proc. Int. Conf. Bus. Process Manage. (BPM)*, in Lecture Notes in Computer Science, vol. 3080, J. Desel, B. Pernici, and M. Weske, Eds. Berlin, Germany: Springer, 2004, pp. 244–260.

[15] R. J. C. Bose, E. H. M. Verbeek, and W. M. P. van der Aalst, "Discovering hierarchical process models using ProM," in *IS Olympics: Information Systems in a Diverse World*. Berlin, Germany: Springer, 2012, pp. 33–48.

[16] R. J. C. Bose and W. M. P. van der Aalst, "Context aware trace clustering: Towards improving process mining results," in *Proc. SDM*, 2009, pp. 401–412.

[17] A. K. A. de Medeiros *et al.*, "Process mining based on clustering: A quest for precision," in *Business Process Management Workshops*. Berlin, Germany: Springer, 2008, pp. 17–29.

[18] J.-Y. Jung, "PROCL: A process log clustering system," *J. Soc. e-Bus. Stud.*, vol. 13, no. 2, pp. 181–194, 2008.

[19] D. Ferreira, M. Zacarias, M. Malheiros, and P. Ferreira, "Approaching process mining with sequence clustering: Experiments and findings," in *Business Process Management*. Berlin, Germany: Springer, 2007, pp. 360–374.

[20] C. Di Francescomarino, A. Marchetto, and P. Tonella, "Cluster-based modularization of processes recovered from web applications," *J. Softw., Evol. Process*, vol. 25, no. 2, pp. 113–138, 2013.

[21] A. Bogarín, C. Romero, R. Cerezo, and M. Sánchez-Santillán, "Clustering for improving educational process mining," in *Proc. 4th Int. Conf. Learn. Anal. Knowl.*, Indianapolis, Indiana, 2014, pp. 11–15.

[22] J. De Weerdt, S. vanden Broucke, J. Vanthienen, and B. Baesens, "Active trace clustering for improved process discovery," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 12, pp. 2708–2720, Dec. 2013.

[23] K. Kim, J.-Y. Jung, and J. Park, "Discovery of information diffusion process in social networks," *IEICE Trans. Inf. Syst.*, vol. E95-D, no. 5, pp. 1539–1542, 2012.

[24] K. Kim, J. Obregon, and J.-Y. Jung, "Analyzing information flow and context for Facebook fan pages," *IEICE Trans. Inf. Syst.*, vol. 97, no. 4, pp. 811–814, 2014.

[25] B. Carrera, J. Lee, and J.-Y. Jung, "Discovering information diffusion processes based on hidden Markov models for social network services," in *Asia Pacific Business Process Management* (Lecture Notes in Business Information Processing), vol. 219, J. Bae, S. Suriadi, and L. Wen, Eds. Cham, Switzerland: Springer, 2015, pp. 170–182.

[26] D. Li, S. Zhang, X. Sun, H. Zhou, S. Li, and X. Li, "Modeling information diffusion over social networks for temporal dynamic prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 9, pp. 1985–1997, Sep. 2017.
[27] C. Jiang, Y. Chen, and K. J. R. Liu, "Evolutionary dynamics of information diffusion over social networks," *IEEE Trans. Signal Process.*, vol. 62, no. 17, pp. 4573–4586, Sep. 2014.
[28] Y. Hu, R. J. Song, and M. Chen, "Modeling for information diffusion in online social networks via hydrodynamics," *IEEE Access*, vol. 5, pp. 128–135, 2017.
[29] K. Zhang, J. Wang, C. Jiang, Z. Wei, and Y. Ren, "Big data driven information diffusion analysis and control in online social networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.
[30] J. Wang, C. Jiang, T. Q. S. Quek, X. Wang, and Y. Ren, "The value strength aided information diffusion in socially-aware mobile networks," *IEEE Access*, vol. 4, pp. 3907–3919, 2016.
[31] V. Arnaboldi, M. Conti, M. La Gala, and A. Passarella, and F. Pezzoni, "Information diffusion in OSNs: the impact of nodes' sociality," in *Proc. 29th Annu. ACM Symp. Appl. Comput.*, Gyeongju, South Korea, 2014, pp. 616–621.

JOSUE OBREGON received the master's degree in industrial and management systems engineering from Kyung Hee University (KHU), South Korea, where he is currently pursuing the Ph.D. degree with the Department of Industrial and Management Systems Engineering, Faculty of Engineering. His research interests include process mining, social networks, machine learning, deep learning, and decision support systems.

MINSEOK SONG received the Ph.D. degree in industrial and management engineering from the Pohang University of Science and Technology (POSTECH), in 2006, where he is currently an Associate Professor with the Department of Industrial and Management Engineering. He was with the Information Systems Group, Technology Management Department, Eindhoven University of Technology, as a Postdoctoral Researcher, from 2006 to 2009. Also, he was an Assistant/Associate Professor with Ulsan National Institute of Science and Technology (UNIST). He has published more than 60 scientific papers in several top-level venues, such as *Decision Support Systems*, *Information Systems*, the *Journal of Information Technology*, and the *International Journal of Medical Informatics*. His research interests include business process management, process mining, business analytics, simulation, and social network analysis.

JAE-YOON JUNG received the B.S., M.S., and Ph.D. degrees in industrial engineering from Seoul National University, in 1999, 2001, and 2005, respectively. He was with the Information Systems Group, Eindhoven University of Technology (TU/e), as a Postdoctoral Researcher, from 2006 to 2007. He has been a Professor with the Department of Industrial and Management Systems Engineering, Kyung Hee University (KHU), since 2007, where he is currently the Director of the Industrial Artificial Intelligence Laboratory. His research interests include process mining, machine learning, smart manufacturing, and industrial artificial intelligence.