Discovering Audience Groups and Group-Specific Influencers

Shuyang Lin[1], Qingbo Hu[1] Jingyuan Zhang[1], and Philip S. Yu[1][2]

[1] University of Illinois at Chicago, Illinois, USA
{slin38,qhu5,jzhan8,psyu}@uic.edu,
[2] Institute for Data Science, Tsinghua University, Beijing, China

Abstract. Recently, user influence in social networks has been studied extensively. Many applications related to social influence depend on quantifying influence and finding the most influential users of a social network. Most existing work studies the global influence of users, i.e. the aggregated influence that a user has on the entire network. It is often overlooked that users may be significantly more influential to some audience groups than others. In this paper, we propose *AudClus*, a method to detect *audience groups* and identify *group-specific influencers* simultaneously. With extensive experiments on real data, we show that AudClus is effective in both the task of detecting audience groups and the task of identifying influencers of audience groups. We further show that AudClus makes possible for insightful observations on the relation between audience groups and influencers. The proposed method leads to various applications in areas such as viral marketing, expert finding, and data visualization.

Keywords: Social influence, influencer detection, audience group

1 Introduction

Quantifying influence to find the most influential users in social networks is a fundamental problem of social network studies. Many important applications such as influencer detection and viral marketing rely on this problem. Most existing studies quantify the influence of a user as a globally aggregated influence value. An observation that is often overlooked by these studies is that a social network contains various groups of users, and the strength of influence of a user varies drastically over different groups. On one hand, most users have their influence limited to a small part of the social network. Even the globally most influential users of a social network have their influence concentrated to some specific groups of audience. On the other hand, different groups of users in a social network have their own specific sets of influencers.

Based on this observation, in this paper, we explore group-specific influence. We attempt to (1) detect audience groups and (2) identify influencers of audience groups. The tasks have two major challenges. **First**, to make the results meaningful, audience groups should reflect natural boundaries of influence, i.e. users in the same audience group share similar influencers, while users in different audience groups have different influencers. However, most existing community detection algorithm is not specifically optimized for detecting such audience groups. **Second**, current methods for detecting global influencer do not work well for the task of detecting group-specific influencers. One can certainly consider each audience group as a network and apply existing influencer detection algorithms to each network. This naive approach, however, yields to suboptimal results, because it can only detect influencers within groups, but an influencer of an audience group may actually be outside that group.

To solve these problems, in this paper, we propose **AudClus**, a probabilistic mixture model based method to detect audience groups and group-specific influencers simultaneously. By using information diffusion data, it groups users into different audience groups according to the users who are influential to them, and simultaneously quantifies the influence of users with respect to each audience group. Both the tasks of detecting audience groups and identifying group-specific influencers benefit from the simultaneous inference.

The main contributions of this paper are summarized as follows.

- We propose AudClus, a probabilistic mixture model based method, to capture audience groups and group-specific influence. We design an EM algorithm to infer audience groups and users' influence over audience groups simultaneously.
- AudClus is very flexible in capturing group-specific influence in social networks. It does not rely on the structure of social networks or any specific information diffusion model. It can capture both direct and indirect influence.
- AudClus provides a new tool for analysis and visualization of social influence. It leads to interesting observations and insightful understandings on the structure of influence in social networks. It facilitates applications such as finding experts in specific areas, and targeting specific groups of audience for viral marketing.

2 Preliminary

A social network is often considered as a graph, with users as nodes, and links between users as edges. By considering a social network as a graph, graph clustering or community detection algorithms can be used to detect groups from the social network. For a given social network, different community detection algorithms may lead to substantially different clustering results. In this paper, we are interested in detecting **audience groups**, which reflect users' behaviors of being influenced in information diffusion processes. Specifically, users in the same audience group are influenced by a similar set of influencers, while users in different groups are influenced by different influencers. To serve this purpose well, we propose **AudClus**, a clustering method, which detect groups of users from **information diffusion data**, instead of from social network structure.

When information diffuses in a social network, it is carried by actions of users in the network. An **action** of a user is, for example, posting a tweet in Twitter, or publishing a paper in the citation network. Each action comes with some information. For example, a tweet can talk about some news and events, and a paper can propose or adopt some techniques. Information carried by an action may be introduced into the social network by the current action itself, or it may be introduced by some previous actions and adopted by the current action. If a current action adopts information from some previous actions, we say that information propagates from the previous actions to the current action, and we define it by an **information propagation link** from each previous action to the current action. In this paper, we study the case that the information propagation links are observable in the data. For example, in Twitter, a retweeting or a replying can be considered as a information propagation link from the original tweet to the current tweet, while in the citation network, a citing can be considered as a link from the reference paper to the current paper.

A **diffusion pathway graph** contains a set of actions and the information propagation links between them. Formally, we define a diffusion pathway graph as follows.

Definition 1. *A diffusion pathway graph $D = (\mathcal{A}_D, \mathcal{L}_D)$ is a DAG (directed acyclic graph) of actions. Each action $a_i \in \mathcal{A}_D$ is taken by a user denoted by v_{a_i}. Directed links in $\mathcal{L}_D \subset \mathcal{A}_D \times \mathcal{A}_D$ define the information diffusion links between actions. A directed link $(a_i, a_j) \in \mathcal{L}_D$ means that action a_j is directly influenced by action a_i. Links in $\mathcal{L}_D$ should be acyclic. If $(a_i, a_j) \in \mathcal{L}_D$, we say a_i is a parent of a_j.*

The above definition of diffusion pathway graph is very general and flexible in the sense that it does not make any assumption on the underlying diffusion process. It can therefore be applied to various information diffusion models, and different information diffusion models may add different constraints to this general definition. For example, for an IC model [10, 4], the diffusion pathway graph is actually a forest, since any action can be triggered by only one previous action, while the diffusion pathway graph for a LT model [9, 5] can be any DAG. Besides, a diffusion pathway graph is not limited to describe the diffusion of one single piece of information. When the pieces of information are not explicitly available, it means less effort in preprocessing data. For example, we can directly construct a diffusion pathway graph from a citation network, with papers as actions, and citation relations as information diffusion links. We do not need to extract the pieces of information that is actually spread between papers.

The first goal of AudClus is to detect audience groups based on diffusion pathway graphs. More formally, given a set of users $\mathcal{V}$ and a set of diffusion pathway graphs $\mathcal{D} = \{D_1, \cdots, D_m\}$ with these users, it detects a set of audience groups $\mathcal{C}$, such that each user $u \in \mathcal{V}$ is assigned to a group $c \in \mathcal{C}$. Notice that the setting is different from that of traditional community detection problem: we detect groups of users from diffusion pathway graphs, instead of from the social network. Unlike social networks, nodes in diffusion pathway graphs are actions, not users. The difference in problem setting means traditional community detection algorithms cannot be directly applied to the task of detecting audience groups.

The second goal of AudClus is to identify influencers who are specific to each audience group. To achieve that, AudClus quantifies the influence from each user

	Description		Description
$\mathcal{V}$	Set of all users.	k	Number of audience groups.
$\mathcal{C}$	Set of all audience groups.	n	Number of users.
$\mathcal{D}$	Set of all diffusion pathway graphs.	θ_{vc}	The influence user v has on group c.
$\mathcal{A}_D$	Set of actions in diffusion pathway graph D.	η_{vc}	The conditional probability that v belongs to c.
$\mathcal{L}_D$	Set of links in diffusion pathway graph D.	ϕ_c	The prior probability of group c.
$z(v)$	The audience group user v is assigned to.	q	Transfer rate parameter in influence backtracing method.

Table 1: Notations

$v \in \mathcal{V}$ to each group $c \in \mathcal{C}$, then it can identify users who are the most influential to a specific group. As we will show in the next section, the two goals of AudClus can be achieved simultaneously under a mixture model framework.

We summarize notations in Table 1.

3 The AudClus method

In this section, we introduce a probabilistic mixture model based method to detect audience groups of a social network and to quantify group-based influence for users simultaneously. We will first study a simple case, which we call single-direct case. In the single-direct case, each action is either spontaneous or influenced by exactly one previous action, i.e. each action either has no parent in the diffusion pathway graph or has one single parent, and only the influence from the parent is considered. We will first show a probabilistic mixture model for audience clustering for the single-direct case, and then extend the model for more general cases.

3.1 Audience clustering for the single-direct case

The intuition for audience clustering is that users who are influenced by similar sets of influencers should be assigned to the same group. The proposed clustering method originates from the probabilistic mixture model proposed in [15], which decides the group of a user according to the neighbors whom he is linked to and assigns users who are linked to similar sets of neighbors into the same group. The original model in [15] was designed for undirected graph, but we extend it to make it work with directed graphs such as diffusion pathway graphs. Besides, in the proposed model, the groups of users are decided by their influencers, not by their neighbors.

The basic concepts with regard to the proposed clustering model are as follows. From a set of users $\mathcal{V}$, each user u is assigned to a group $c \in \mathcal{C}$, denoted by $z(u)$. For each group $c \in \mathcal{C}$ and each user $v \in \mathcal{V}$, θ_{vc} defines the influence that user v has on group c. More specifically, for an action taken by users in group c, θ_{vc} is the probability that the action is influenced by some previous actions taken by user v. Notice that we are considering the single-direct case that each action is either spontaneous or influenced by exactly one parent action. For each action a_i that is influenced by a parent action, we regard the user who takes the parent action as the influencer of a_i, denoted by $r(a_i)$. $r(a_i)$ is generated from a categorical distribution as follows.

$$r(a_i) \sim Categorical_{|\mathcal{V}|}(\boldsymbol{\theta}_{\cdot z(v_{a_i})})$$

where v_{a_i} is the user who takes action a_i, and $z(v_{a_i})$ is the group that v_{a_i} belongs to. $\boldsymbol{\theta}_{\cdot c} = \{\theta_{vc}\}_{v\in\mathcal{V}}$ denotes the influence from users in $\mathcal{V}$ to group c.

For any group c, $\boldsymbol{\theta}_{\cdot c}$ are the parameters for a categorical distribution, which should satisfy the following normalization condition.

$$\textstyle\sum_{v\in\mathcal{V}} \theta_{vc} = 1,\ \forall c \in \mathcal{C}$$

We consider the clustering of users as a probabilistic mixture model. The prior probability for group c is denoted by ϕ_c, satisfying the following normalization condition.

$$\textstyle\sum_{c\in\mathcal{C}} \phi_c = 1$$

We denote with $\mathcal{Z}$ the multivariate random variable that consists of $z(v)$ for all $v \in \mathcal{V}$, i.e. $\mathcal{Z} = \{z(v)\}_{v\in\mathcal{V}}$. Similarly, we have $\Theta = \{\theta_{vc}\}_{v\in\mathcal{V},c\in\mathcal{C}}$ and $\Phi = \{\phi_v\}_{v\in\mathcal{V}}$.

Given the parameters Θ and Φ, the joint probability of $\mathcal{D}$ and $\mathcal{Z}$ is the product of two probabilities: the probability that each user v is assigned to the group $z(v)$, and the probability that each action a_i influenced by the influencer $r(a_i)$. Formally, the likelihood function of parameters Θ and Φ, are given as follows.

$$\begin{aligned}\mathcal{L}(\Theta,\Phi;\mathcal{D},\mathcal{Z}) =& \Big(\textstyle\prod_{D\in\mathcal{D}}\prod_{a_i\in\mathcal{A}_D} \theta_{r(a_i)z(v_{a_i})}\Big)\Big(\textstyle\prod_{v\in\mathcal{V}} \phi_{z(v)}\Big)\\ =& \Big(\textstyle\prod_{v\in\mathcal{V}}\prod_{u\in\mathcal{V}} \theta_{vz(u)}^{A_{vu}}\Big)\Big(\textstyle\prod_{v\in\mathcal{V}} \phi_{z(v)}\Big)\end{aligned} \tag{1}$$

where

$$A_{vu} = \textstyle\sum_{D\in\mathcal{D}} \sum_{\substack{a_i\in\mathcal{A}_D,\\ v(a_i)=u}} I_{r(a_i)=v} \tag{2}$$

denotes the number of actions of user u that are influenced by user v in all diffusion pathway graphs.

Parameter estimation. We estimate the parameters Θ and Φ by their maximum likelihood estimation. Notice that the group of each user $z(u)$ is the missing data that also needs to be inferred. Therefore, the problem of finding maximum likelihood estimation is formalized as follows:

$$\max_{\Theta,\Phi} \textstyle\sum_{\mathcal{Z}} p(\mathcal{D},\mathcal{Z}|\Theta,\Phi)$$

We solve this problem by EM algorithm. In the E-step, we calculate expected value of log-likelihood function, with respect to the conditional distribution $\mathcal{Z}$. The expected value is defined as follows.

$$E_{Z|\Theta^{(t)},\Phi^{(t)}} \log \mathcal{L}(\Theta,\Phi;\mathcal{D},\mathcal{Z}) = \sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{V}}\sum_{c\in\mathcal{C}} A_{vu}\eta_{uc}^{(t)} \log\theta_{vc} + \sum_{v\in V}\sum_{c\in\mathcal{C}} \eta_{vc}^{(t)} \log(\phi_c) \tag{3}$$

where

$$\eta_{uc}^{(t)} = \Big(\phi_c^{(t)} \prod_{v\in\mathcal{V}} (\theta_{vc}^{(t)})^{A_{vu}}\Big) / \Big(\sum_{c'\in\mathcal{C}} \phi_{c'}^{(t)} \prod_{v\in\mathcal{V}} (\theta_{vc'}^{(t)})^{A_{vu}}\Big) \tag{4}$$

is the conditional probability that $z(u) = c$ given $\mathcal{D}$ under the current estimations of parameters $\Theta^{(t)}$ and $\Phi^{(t)}$.

In the M-step, we update estimation of Θ and Φ to maximize the expected log-likelihood. By taking partial derivatives of Equation 3, we can find out that the expected log-likelihood is maximized by the following values of parameters.

$$\theta_{vc}^{(t+1)} = \Big(\textstyle\sum_{u\in\mathcal{V}} A_{vu}\eta_{uc}^{(t)} \Big) / \Big(\sum_{w\in\mathcal{V}} \sum_{u\in\mathcal{V}} A_{wu}\eta_{uc}^{(t)} \Big) \tag{5}$$

and

$$\phi_{c}^{(t+1)} = \Big(\textstyle\sum_{u\in\mathcal{V}} \eta_{uc}^{(t)} \Big) / \Big(\sum_{c'\in\mathcal{C}} \sum_{u\in\mathcal{V}} \eta_{uc'}^{(t)} \Big) \tag{6}$$

We repeat the E-step and the M-step until it converges. When it converges, the **influence** of user v on group c is defined by θ_{vc}, the value that $\theta_{vc}^{(t)}$ converges to, while the **belongingness** of user v to group c is defined by η_{vc}, the value that $\eta_{vc}^{(t)}$ converges to. When a non-probabilistic clustering is needed, we assign user v to the group c with the largest η_{vc}, i.e. $z(v) = \arg\max_{c\in\mathcal{C}} \eta_{vc}$.

3.2 Generalized model

In the previous section, we have introduced the audience clustering algorithm for the single-direct case that each action either has no parent or has one single parent in the cascade, and only the influence from the parent is considered. Many real applications, however, do not satisfy this condition for two reasons. First, in many diffusions, each action may have multiple parent actions. For example, in the citation network, each action (paper) can cite multiple previous papers, thus has multiple parent actions. Second, in many applications, both direct and indirect influence is important and should be considered. For example, in the citation network, an influential paper should not only have a large citation number itself, but also inspires some innovative papers which also have plenty of citations.

We first propose a partial credit method to generalize the clustering method, so that actions can have arbitrary number of parents in the diffusion pathway graph, and then further propose a influence backtracking method to incorporate both direct and indirect influence under the same model.

Partial credit method In the single-direct case, each action a_i has one single influencer $r(a_i)$. If each action can have multiple parents, the assumption will be violated. However, similar to the partial credit method in [9], we can generalize the model by letting all parents of an action share the "credit" of influencing that action. Notice that the likelihood function in Equation 1 actually depends on A_{vu}, the number of times that user v influences user u. Thus, we can replace A_{vu} in Equation 2 with following value.

$$A_{vu} = \textstyle\sum_{D\in\mathcal{D}} \sum_{\substack{a_j\in\mathcal{A}_D,\\ v(a_j)=v}} \sum_{\substack{a_i\in\mathcal{A}_D,\\ v(a_i)=u}} \frac{1}{|F(a_i)|} I_{a_j\in F(a_i)} \tag{7}$$

where $F(a_i)$ is set of parents of action a_i. In this equation, each parent of action a_i gets $\frac{1}{|F(a_i)|}$ credit for influencing action a_i.

Influence backtracking method We now introduce an influence backtracking method to measure the influence between actions in a diffusion pathway graph. The benefits of influence backtracking method are as follows: (1) the same as the partial credit method, each action can have arbitrary number of parents; (2) both direct and indirect influence is captured by the same measurement. By incorporating the influence backtracking method, the AudClus model can be generalized to all diffusion pathway graphs under the flexible definition as in Definition 1.

The intuition of influence backtracking is measuring influence by the amount of information that is brought into the social network by an action and is adopted by following actions. Consider the scenario of an author writing a blog post. The author gets some information from some other blog posts, and brings in some new ideas at the same time. Therefore, some information in this new blog post originates from previous blog posts, which are listed as references of this post, and some may be traced back even further to references of the references. For each piece of information carried by action a_i, we can trace back the diffusion pathway graph to find out which action brings that piece of information to the network.

To simulate this process, for each piece of information in an action a_i, we use a reverse random walk to trace back which action brings this piece of information to the network. The reverse random walk is defined as follows:
1. It starts at the node a_i.
2. When it arrives at a node a_j with no parents, the random walk terminates at a_j.
3. When it arrives at a node a_j with some parents, with probability $1-q$ the random walk terminates at the node a_j, and with probability q the random walk continues. If it continues, it has equal probability to walk to each parent of a_j.
If the random walk terminates at the action a_j, it represents that the piece of information originates from action a_j. We call q the transfer rate parameter of influence backtracking.

Since the diffusion pathway graph is an acyclic graph of actions, it is easy to calculate the probability that the random walk terminates at action a_j. The probability can be calculated recursively by the equation as follows.

$$Q(a_j, a_i) = \begin{cases} 1 & \text{if } F(a_i) = \emptyset,\ i = j \\ 0 & \text{if } F(a_i) = \emptyset,\ i \neq j \\ 1 - q & \text{if } F(a_i) \neq \emptyset,\ i = j \\ \frac{q}{|F(a_i)|} \sum_{a_k \in F(a_i)} Q(a_j, a_k) & \text{if } F(a_i) \neq \emptyset,\ i \neq j. \end{cases}$$

where $F(a_i)$ is set of parents of a_i.

Suppose that there are M pieces of information in an action a_i. We use the random walk to trace back where each piece of information is from. When M is large enough, the fraction of information pieces that are carried by action a_i and originate from action a_j is approximately $Q(a_j, a_i)$. Thus, for an action a_i and a previous action a_j, we regard $Q(a_j, a_i)$ as the part of a_i that is influenced by a_j, and replace A_{vu}, the number of actions of user u that are influenced by actions of v in Equation 2, by the value as follows.

$$A_{vu} = \sum_{D \in \mathcal{D}} \sum_{\substack{a_j \in \mathcal{A}_D,\\ v(a_j)=v}} \sum_{\substack{a_i \in \mathcal{A}_D,\\ v(a_i)=u}} Q(a_j, a_i) \tag{8}$$

4 Experiment

4.1 Experiment Setup

Datasets We experiment with two real-world datasets.

– **Citation dataset** This is the citation network dataset released by Arnet-Miner [17]. The publication data are extracted from DBLP, ACM and other sources. Since the influence of authors changes over time, to reflect current landscape of influence, we have removed publications before year 2000. We have also filtered out authors who have less than 5 publications. After preprocessing, the dataset contains 368,101 publications of 113,006 authors, and 592,889 citation links. It also contains conference information of publications, which we use for clustering evaluation. The original dataset considers venues such as CoRR as "conferences". It also contains some less competitive conferences which have far more larger number of accepted papers than normal conferences. We use the conference list provided by Microsoft academic search (http://academic.research.microsoft.com/) to clean the data. We only consider the top 100 computer science conferences listed by Microsoft academic search in the experiment.

– **Meme dataset** This is the meme dataset from Memetracker [11]. It contains posts from news websites and blogs, and links between then. We use the meme dataset crawled at August 2008. We consider websites as nodes ("users") of the network, and posts in websites as actions. Diffusion pathway graphs are generated from links between posts. We have removed websites with less than 5 posts. After preprocessing, the dataset contains 40,072 websites, 394,636 pages, and 1,394,710 links.

Methods For quantitative evaluation, we compare following proposed methods and baselines.

– **AC-i** AudClus with information backtracking.

– **AC-p** AudClus with partial credit.

– **MD** Mixture model proposed in [15]. We construct a directed network of users by adding a directed link from user v to user u, if there is some actions of user u linked to some actions of user v in some diffusion pathway graphs.

– **FG** Fast greedy algorithm in [6]. It is a modularity based community detection algorithm. The algorithm is implemented in the igraph network analysis package [7]. Similar to **MD**, we construct a directed network of users from diffusion pathway graphs, but each edges in the user network weighted by the number of links between actions of the two users.

4.2 Qualitative analysis and case studies

Citation dataset. We begin with a qualitative analysis on the citation dataset. As we will show later, **AC-i** achieves best quantitative evaluation result among all algorithms, and it achieve best result with $q = 0.3$ and $k = 35$ for the citation dataset. We conform to this setting for this part of experiment.

In Table 2, we show an overview of the audience groups. For each group, we show its top 3 most common conferences, and its top one influencer. We assign each user v to the group that he has the largest belongingness η_{vc}. We identify frequent conferences of a group by counting the number of users in that group who have published papers in each conference, and finding the top 3 conferences with the largest counts. We identify the top influencer of a group by finding the user with the largest influence θ_{vc}.

The list of top frequent conferences for groups provides intuitive observations on the clustering quality. First, the frequent conferences for a certain group are usually conferences that are related to the same research area. For example, the top 3 frequent conferences for group 5 (KDD, CIKM, ICDE) are conferences related to the data mining area, while those for group 8 (ICIP, CVPR, ICCV) are related to the computer vision area. Second, different groups tend to have different list of frequent conferences. No two groups share exactly the same frequent conference list. Most overlaps of frequent conferences between groups can be explained by the phenomenon that a conference is often related to more than one research areas. For example, the top 3 frequent conferences of group 3 are ICDE, VLDB and CIKM, and CIKM and ICDE are also the 2nd and 3rd most frequent conferences for group 5, respectively. The explanation for it is as follows: (1) group 3 reflects the database area, while group 5 reflects the data mining area. (2) CIKM and ICDE accept papers in both the database and data mining areas.

To get a detailed observation on the extracted audience groups, we show longer lists for top influencers and top frequent conferences for group 5 in Table 5. Values in parentheses after names of influencers are θ_{vc} for that influencer. Values in parentheses after conference are the percentage of users in the group who have publications in the conference. It is very obvious that the top frequent conferences are related to the data mining area and the top influencers are indeed influential researchers in that area.

In Figure 1, we display another case study. In Figures 1(a) and 1(b), we show the influence spread (θ_{vc}) and belongingness distribution (η_{vc}) of author *Jiawei Han* over all groups. As shown by the figures, the belongingness of Jiawei Han almost completely concentrates to group 5. However, the influence of Jiawei Han spreads over several groups, although his influence on group 5 is much larger than his influence on other groups. Similar observations hold for other users in the network as well: a user who is influential to a group does not have to be a member of that class; a user can have influence over more than one groups; however, the strength of influence can be very different in different groups.

Meme dataset. For the meme dataset, we use **AC-i** with $q = 0.3$ and $k = 10$. Since there is no information like conferences in the citation network, it

	Most common conf.	Top Influencer
0	SC, ICS, ICPP	Jos E. Moreira
1	ICC, WCNC, ICASSP	Anil K. Jain
2	ICSE, HICSS, ECOOP	Barry W. Boehm
3	ICDE, VLDB, CIKM	David J. DeWitt
4	PODC, ICDCS, ICC	Sanjay Jain
5	KDD, CIKM, ICDE	Jiawei Han
6	CDC, EUROCRYPT,CRYPTO	Moni Naor
7	SODA, ICRA, ICIP	Joseph S. B. Mitchell
8	ICIP, CVPR, ICCV	David J. Hawkes
9	CAV, CONCUR, LICS	Moshe Y. Vardi
10	IJCAI, AAAI, KR	Endre Boros
11	NIPS, ICML, IROS	Kalyanmoy Deb
12	ICRA, IROS, CHI	David H. Laidlaw
13	DAC, ICCAD, DATE	David Blaauw
14	ACL, SIGIR, COLING	Andrew McCallum
15	IJCAI, HICSS, AAAI	Gheorghe Paun
16	CHI, CSCW, UIST	Benjamin B. Bederson
17	ICIP, ICPR, ICASSP	Anil K. Jain
18	ITC, DATE, DAC	Krishnendu Chakrabarty
19	ICRA, IROS, ICPR	Sebastian Thrun
20	DATE, DAC, ISCAS	Margaret Martonosi
21	HPDC, SC, ICPP	Ian T. Foster
22	IJCAI, AAAI, ICALP	Jack H. Lutz
23	AAAI, IJCAI, ICRA	Milind Tambe
24	SC, POPL, LICS	William Gropp
25	ICC, IROS, INFOCOM	Edward R. Dougherty
26	ICASSP, ISCAS, ICC	Aapo Hyvrinen
27	WWW, CIKM, ICDE	Ian Horrocks
28	HICSS, ICC, ICRA	Viswanath Venkatesh
29	CDC, ICC, HICSS	Wil M. P. van der Aalst
30	ICC, INFOCOM, WCNC	Donald F. Towsley
31	ICIP, ICPR, ICC	Etienne E. Kerre
32	ICASSP, ACL, IROS	Andreas Stolcke
33	OR, SODA, ICC	David E. Goldberg
34	SODA, STOC, FOCS	Christos H. Papadimitriou

Table 2: Audience groups

	AC-i	**AC-p**	**MD**	**FG**
Preprocessing time	263.5	123.2	69.9	77.4
Inference time	104.94	39.5	34.3	7182.4

Table 3: Running time of algorithms (in seconds)

	First group(political news)	
	website	description
1	telegraph.co.uk	newspaper
2	foxnews.com	news channel
3	nydailynews.com	newspaper
4	msnbc.msn.com	news channel
5	timesonline.co.uk	newspaper
6	nypost.com	newspaper
7	news.bbc.co.uk	news channel
8	cnn.com	news channel
9	politicalticker.blogs.cnn.com	political news blog
10	troktiko.blogspot.com	political news blog
	Second group (technology)	
	website	description
1	blog.wired.com	technology magazine
2	gizmodo.com	technology blog
3	digg.com	news aggregator (technology)
4	telegraph.co.uk	newspaper
5	arstechnica.com	technology news
6	universetoday.com	technology news
7	sciencedaily.com	technology news
8	engadget.com	technology blog
9	msnbc.msn.com	news channel
10	scienceblogs.com	technology blog

Table 4: Case study: top influential websites

	Top influencers	Frequent conference
1	Jiawei Han (0.0251)	KDD (38.5%)
2	Jian Pei (0.0194)	CIKM (33.0%)
3	Charu C. Aggarwal (0.0096)	ICDE (29.2%)
4	Mohammed Javeed Zaki (0.0075)	VLDB (15.2%)
5	Philip S. Yu (0.0060)	WWW (11.8%)
6	Ramesh C. Agarwal (0.0060)	AAAI (9.4%)
7	Johannes Gehrke (0.0035)	ICML (8.8%)
8	Ke Wang (0.0043)	IJCAI (8.7%)
9	George Karypis (0.0041)	SIGIR (8.0%)
10	Rakesh Agrawal (0.0035)	ICC (6.0%))

Table 5: Case study: data mining group (group 5)

is hard to summarize the category or area of each group. Instead, we show a case study with that dataset. In Table 4, we show the top influencer of two groups. For the first group, the top 8 influencers are either newspapers or news channels, and political news is a major topic covered by these newspapers and channels. The last 2 are political news blogs. For the second group, most of the top influencers are technology news websites, except for msnbc.msn.com. Notice that msnbc.msn.com appears in the top influencer lists of both groups, but it is more influential on the first group than on the second group. It conforms to the intuition that MSNBC is a more influential source for political news than it is for technology news.

4.3 Quantitative analysis

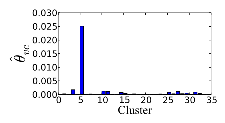
Clustering evaluation In this section, we evaluate the quality of clustering quantitatively. For both datasets, the ground truth of groups is not available. However, for the citation dataset, we can use the conferences of an author's papers to roughly identify the group of authors. Thus, we can conduct quantitative evaluation of clustering on the *citation dataset*.

We use three measurements for the evaluation of clustering: purity, mutual information (MI), and normalized mutual information (NMI). The three measurements are often used for evaluation of clustering quality. They are usually defined for datasets with a set of explicit classes and each node is assigned to exactly one class. In the citation datasets, conferences of authors' identify the areas of study and can be used for the evaluation. However, unlike datasets with explicit classes, each author in the citation dataset can publish in multiple conferences. To make the evaluations work for the citation dataset, we use modified definitions of purity, MI, and NMI as follows:

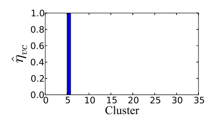
- **purity** We first find most frequent conferences for each group as we did in the last section, and then defined $purity(m)$ as the fraction of users who have published in at least one of the top m most frequent conferences of their groups.
- **MI** For each conference e, we divide users into positive class (users who published in this conference), and negative class (users who did not publish in this conference). We then calculate the mutual information between positive/negative classes and groups of users. Formally, the mutual information is defined as $I(f, \mathcal{C}) = \sum_{l_e=\{0,1\}} \sum_{c \in \mathcal{C}} \frac{N_{l_e c}}{N} \log \frac{N_{l_e c} N}{N_{l_e \cdot} N_{\cdot c}}$, where $l_e = 1$ and $l_e = 0$ represent the positive and negative classes, respectively. $N_{l_e c}$ is the number of users in group c that belong to the positive or negative class. $N_{l_e \cdot} = \sum_{c \in \mathcal{C}} N_{l_e c}$ is the total number of users belong to the positive or negative class. $N_{\cdot c} = \sum_{l_e \in \{0,1\}} N_{l_e c}$ is the total number of users in group c. N is the total number of users. For each conference e, we calculate the mutual information according to the equation above, and then we calculate the average value over all conferences.
- **NMI** When the number of groups k is large enough, it is easy for a clustering method to achieve high values of purity and MI. Normalized mutual information (NMI) can be used to tradeoff the clustering quality against the increasing number of groups. For a conference e, the normalized mutual information between it and clustering $\mathcal{C}$ is defined as $NMI(e, \mathcal{C}) = I(e, \mathcal{C}) / \sqrt{H(e) H(\mathcal{C})}$, where $H(e) = -\sum_{l_e=\{0,1\}} \frac{N_{l_e \cdot}}{N} \log(\frac{N_{l_e \cdot}}{N})$ is the entropy for conference e, and $H(\mathcal{C}) = -\sum_{c \in \mathcal{C}} \frac{N_{\cdot c}}{N} \log(\frac{N_{\cdot c}}{N})$ is the entropy for clustering. Similar to MI, we use the average value over all conferences to evaluate the clustering.

Since the result clusterings of **AC-i**, **AC-p** and **MD** are influenced by the random values we use to initialize parameters, we run each of them 10 times for each setting and show both mean value and standard deviation.

Figure 3 illustrates purity evaluation for all algorithms (for **AC-i**, q is set to 0.3). In Figures 3(a), 3(b) and 3(c), we show $purity(m)$ with $m = 1, 3, 5$, respectively, for varying number of groups. In each case, when k increases, $purity(m)$ for **AC-i**, **AC-p** and **MD** first increases with increasing k, and then stays around a certain value or even drops slightly when k is large. For **FG**, $purity(m)$ increases slightly when k increases. For each case, **AC-i** and **AC-p** consistently outperform **MD**. It suggests that by considering diffusion pathway graphs of actions, the proposed mixture model based algorithms **AC-i** and **AC-p** improves over **MD**, which is also a mixture model based algorithms but considers the

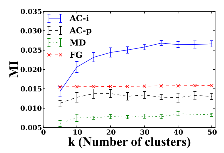


(a) Influence

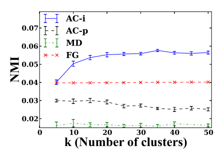


(b) Belongingness

Fig. 1: Case study: Jiawei Han

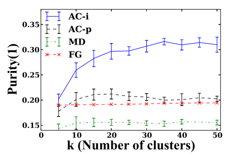


(a) MI

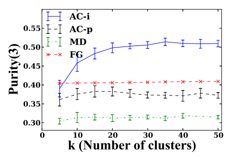


(b) NMI

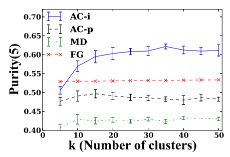
Fig. 2: MI and NMI



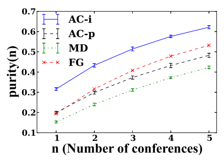
(a) purity(1)



(b) purity(3)



(c) purity(5)



(d) purity(m)

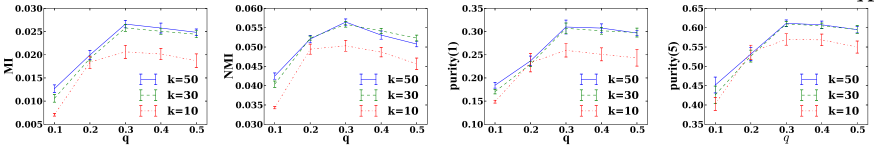
Fig. 3: Clustering evaluation with purity

links between users only. Further more, **AC-i** achieves better clustering quality than **AC-p** and **MD**. That is because **AC-i** quantifies both the direct and indirect influence simultaneously, while **AC-p** and **MD** consider direct influence only. Clustering quality of **AC-p** and **FG** are comparable, with **AC-p** slighter better for *purity*(1), and **FG** better for *purity*(3) and *purity*(5). This is is more clearer in Figure 3(d), in which we show *purity*(m) with varying m for $k = 35$. As illustrated Figure 3(d), when m, the number of top conferences of each group, increases, *purity*(m) for all algorithms goes up steadily. For each m, **AC-i** always achieves best *purity*(m) among four algorithms, while **MD** always has lowest *purity*(m). **AC-p** and **FG** has similar *purity*(m) for $m = 1$. When n increases, **FG** achieves better clustering quality than **AC-p**. However, as we will show later, **FG** is much slower than mixture model based algorithms (**AC-p**, **AC-i**, and **MD**).

Figure 2(a) illustrates the MI measurement for algorithms. In the figure, k is illustrated on the X-axis, while MI is illustrated on the Y-axis. As shown by the figure, when k increases, the MI for **AC-i**, **AC-p** and **MD** first increases and then stays stable. while the MI for **FG** only slightly increases as k increases from 5 to 50. **AC-i** achieves best clustering quality among the three algorithms, and **AC-p** and **FG** outperform **MD**.

Figure 2(b) illustrates the NMI measurement. Comparing with the MI measurement in Figure 2(a), the normalization makes NMI favors clustering with smaller k. For **AC-i**, the maximum of NMI is achieved at $k = 35$, while for **AC-p**, it is achieved at $k = 15$. Comparing with the curve of **FG** in Figure 2(a), we can find out that the clustering of **FG** actually has an almost fixed entropy $H(\mathcal{C})$ when k increases. That suggests that **FG** does not make fully use of increasing number of groups k, and assign most of users to a handful of groups.

Running time Table 3 lists the running time for four algorithms. The running time for each algorithm has two parts: the preprocessing time and the inference time. For **AC-i** and **AC-p**, the preprocessing time is the time spent on calculating A_{vu} for all pair of users. For **MD** and **FG**, the preprocessing time is the time spent on constructing user networks. For each algorithm, the inference time



(a) MI (b) NMI (c) purity(1) (d) purity(5)

Fig. 4: Selection of parameter q

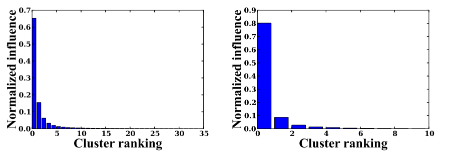
is the time spent on generating the clustering. In Table 3, we show the preprocessing time and the inference time for the citation network for the case $k = 50$. As shown in the table, comparing with **MD**, both the preprocessing time and the inference time of **AC-i** and **AC-p** are increased as they consider actions for clustering inference. **AC-i** takes more time than **AC-p** because both direct and indirect influences are considered by **AC-i**. Although the preprocessing time of **FG** is similar to that of **MD**, the inference time is significantly longer than those of other algorithms. That is because, as a modularity-based community detection algorithm, **FG** invokes time-consuming calculation on graphs. Nevertheless, as we showed in the previous section, the clustering quality of **AC-p** is similar to **FG**, and the clustering quality of **AC-i** is significantly better than that of **FG**.

Selection of parameter q. The clustering quality of **AC-i** depends on the transfer rate parameter q. In Figure 4, we illustrate the clustering quality of **AC-i** with varying q. Figures 4(a), 4(b),4(c) and 4(d) show the clustering evaluation with MI, NMI, *purity*(1), and *purity*(5), respectively. For each measurement, we show the curves for $k = 10$, 30 and 50. In each case, when q varies from 0.1 to 0.5, the clustering quality first increases then decreases. The explanation is as follows: when q is too small, the indirect influence is underestimated; when q is too large, the indirect influence is overestimated by the algorithm. For each case, **AC-i** achieves the best clustering quality when $q = 0.3$.

4.4 Observations on group-specific influence

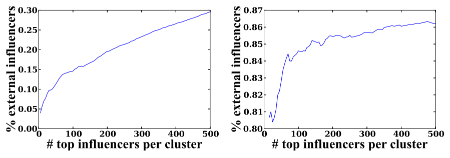
Influence spread of users. First, we study how the influence of users spreads over groups. To illustrate that, we first normalize the influence of each user v over groups with his overall influence, i.e. $\theta_{vc} / \sum_{c' \in \mathcal{C}} \theta_{vc'}$, and rank groups according to the normalized influence v has on the groups. We then show the average normalized influence users have on their first ranked groups, second ranked groups, etc. As illustrated by the figure, the influence spread of users tends to concentrate to the first ranked groups. For the citation dataset, on average, more than 65% of a user's influence concentrates to the first ranked group. Nevertheless, users can still have significant influence on a few other groups. For example, in the citation dataset, on average, the second ranked group for a user has 15% influence of that user. These results confirm the intuition that the strength of influence of a user varies drastically over audience groups, but the influence is not limited to the group that the user belongs to.

The second question we are interested in is whether the overall influence of users is correlated with the extent that their influence spreads on different



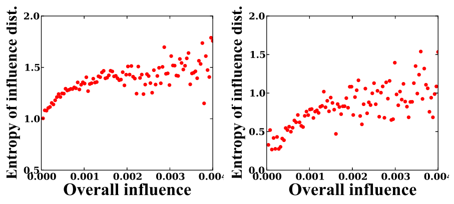
(a) Citation (b) Meme

Fig. 5: User influence spread over groups



(a) Citation (b) Meme

Fig. 6: Fraction of external influencers



(a) Citation (b) Meme

Fig. 7: Entropy of influence spread distribution vs. overall influence

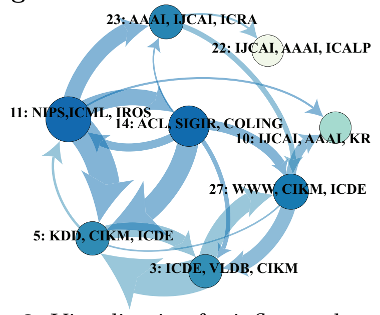


Fig. 8: Visualization for influence between some groups in the citation dataset

groups. To answer this question, we first quantify the extent of influence spread of a user by the entropy of user influence distribution. With larger entropy, the influence of that user tends to spread over different groups more evenly. In Figure 7, we illustrate the entropy of influence spread for users with increasing overall influence. Each point in the figure represents a small range of user overall influence, illustrated on the X-axes. The average entropy for users whose overall influence is within that range is illustrated on the Y-axes. As shown by the figure, comparing with users who has larger overall influence, users with smaller overall influence are more likely to have their influence concentrated to fewer groups.

Fraction of external influencers. The other question about the influencers of groups that we study is: given an audience group, among the top m influencers of the group, how many of them belong to this group, and how may of them belong to other groups. Figure 6 illustrates the fraction of external influencers. In the figure, X-axes illustrate m, the number of top influencers of a group, while Y-axes illustrate the fraction of external influencers, i.e. the fraction of top influencers who does not belong to the group. (We show average value of that fraction over all groups). For both the citation and meme datasets, the fraction of external influencers increases as m increases, which suggests that influencers with larger influence on a group are more likely to be a member of that group. Moreover, the fraction of external influencers is larger in the meme dataset than in the citation dataset. The difference can be explained by the inherent difference between citation networks and website networks: academic authors usually specialize in one or a few areas and seldom have large influence outside the area they specialize in, while many influential websites are comprehensive websites that cover various topics.

Visualization of influence between groups At last, as an example for possible applications of AudClus in influence visualization, in Figure 8, we show the influence between 7 groups in the citation dataset. To quantify the influence from group c_i to group c_j, we calculate the average influence users in c_i have on group c_j, i.e. $Inf(c_i, c_j) = \sum_{z(v)=c_i} \theta_{vc_j}/|N_{c_i}|$ where N_{c_i} is the total number of users in c_i. We selected the groups that are related to database, data mining, and machine learning areas. For each group, we show its index, as well as the top 3 most frequent conferences. The complicated relation between those areas is clearly illustrated by the figure. For example, the research area of data mining (group 5) is strongly influenced by the areas of database (group 3), natural language processing (group 11), and machine learning theory (group 14). On the other hand, the area of data mining (group 5) also has large influence back to the database area (group 3), while it has less influence to the natural language processing and machine learning theory areas.

5 Related work

Quantifying Influence. There has been extensive work on the problem of quantifying influence and detecting the most influential users. Some work regarded influence as the outcome of information diffusion processes, like the independent cascade (IC) model [10, 4] and the linear threshold (LT) model [9, 5]. This line of work proposed methods for finding a set of users such that the expected influence is maximized under a given information diffusion model. Another line of work conducted empirical studies to quantify the influence of users [3, 1]. Topic-dependent influence was also studied frequently in recent years [13, 16].
Information diffusion and community. Recently, researchers have taken notice of the relation between information diffusion and community structures in social networks. [18, 12] proposed community-based greedy algorithms to speed up influence maximization on social networks. [8] generalized the influence maximization problem to the group level. Latest work in [14, 2] analyzed social influence on the community level, which were closely related to our work in this paper. [14] proposed a hierarchical method to summarize social influence by reciprocal influence strength between communities. [2] proposed a stochastic mixture membership generative model to detect cascade-based community. The tasks that we work on are different from [14, 2] in following aspects. **First**, our method focuses on detecting audience groups, while the previous models grouped users based on how they influence others and how they are influenced by others mixedly. **Second**, our method captures the influence of each user to each audience group. Therefore, it works for the task of identifying the most influential users to groups. The previous models only captured the influence between groups, and could not be used to identify the most influential users.

Since influence between users is inherently asymmetric, most traditional community detection algorithms designed for undirected network do not work well for the study of social influence. [15] proposed a community detection method based on probabilistic mixture model. It is a very flexible model that can be naturally extended to detect role-based groups, such as audience groups.

6 Conclusion

In this paper, we study audience groups in networks and group-specific influence of users. We propose AudClus, a mixture model based algorithm to detect audience groups and quantify group specific influence simultaneously. We also invent an influence backtracking method to capture both direct and indirect influence. We show qualitative and quantitative evaluations on real-world datasets. The proposed AudClus algorithm provides a new approach to understand the structure of influence in social networks, which leads to many insightful observations.

Acknowledgements. This work is supported in part by NSF through grants CNS-1115234, Google Research Award, the Pinnacle Lab at Singapore Management University, and Huawei grants.

References

1. E. Bakshy, J. M. Hofman, D. J. Watts, and W. A. Mason. Everyone s an Influencer: Quantifying Influence on Twitter. In *WSDM*, 2011.
2. N. Barbieri, F. Bonchi, and G. Manco. Cascade-based community detection. In *WSDM*, 2013.
3. M. Cha, H. Haddadi, F. Benevenuto, and K. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *ICWSM*, 2010.
4. W. Chen and Y. Wang. Efficient influence maximization in social networks. In *KDD*, 2009.
5. W. Chen, Y. Yuan, and L. Zhang. Scalable Influence Maximization in Social Networks under the Linear Threshold Model. In *ICDM*, 2010.
6. A. Clauset, M. E. J. Newman, and C. Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70(6 Pt 2):066111, 2004.
7. G. Csardi and T. Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems:1695, 2006.
8. M. Eftekhar, Y. Ganjali, and N. Koudas. Information cascade at group scale. In *KDD*, 2013.
9. A. Goyal, F. Bonchi, and L. V. Lakshmanan. Learning influence probabilities in social networks. In *WSDM*, 2010.
10. D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.
11. J. Leskovec, L. Backstrom, and J. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD*, 2009.
12. S. Lin, Q. Hu, G. Wang, and P. S. Yu. Understanding community effects on information diffusion. In *PAKDD*, 2015.
13. L. Liu, J. Tang, J. Han, M. Jiang, and S. Yang. Mining topic-level influence in heterogeneous networks. In *CIKM*, 2010.
14. Y. Mehmood and N. Barbieri. CSI: Community-level social influence analysis. In *ECML/PKDD (2)*, 2013.
15. M. E. J. Newman and E. A. Leicht. Mixture models and exploratory analysis in networks. *PNAS*, 104(23):95649569, 2007.
16. J. Tang, J. Sun, C. Wang, and Z. Yang. Social influence analysis in large-scale networks. In *KDD*, 2009.
17. J. Tang, J. Zhang, L. Yao, J. Li, L. Zhong, and Z. Su. Arnetminer: extraction and mining of academic social networks. In *KDD*, 2008.
18. Y. Wang, G. Cong, G. Song, and K. Xie. Community-based Greedy Algorithm for Mining Top-K Influential Nodes in Mobile Social Networks. In *KDD*, 2010.