User Profiling in an Ego Network: Co-profiling Attributes and Relationships [*]

Rui Li[†], Chi Wang[†], Kevin Chen-Chuan Chang[†,‡]
{ruili1, chiwang1, kcchang}@illinois.edu
[†] Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA
[‡] Advanced Digital Sciences Center, Illinois at Singapore, Singapore

ABSTRACT

User attributes, such as occupation, education, and location, are important for many applications. In this paper, we study the problem of profiling user attributes in social network. To capture the correlation between attributes and social connections, we present a new insight that social connections are discriminatively correlated with attributes via a *hidden* factor – relationship type. For example, a user's colleagues are more likely to share the same employer with him than other friends. Based on the insight, we propose to *co-profile* users' attributes and relationship types of their connections. To achieve co-profiling, we develop an efficient algorithm based on an optimization framework. Our algorithm captures our insight effectively. It iteratively profiles attributes by propagation via certain types of connections, and profiles types of connections based on attributes and the network structure. We conduct extensive experiments to evaluate our algorithm. The results show that our algorithm profiles various attributes accurately, which improves the state-of-the-art methods by 12%.

Categories and Subject Descriptors

H.2.8 [**Data Management**]: Database Applications - Data Mining; H.4.0 [**Information Systems**]: General

Keywords

Social Network; Ego Network; User Profiling

1. INTRODUCTION

User profiling, which refers to the process of inferring users' essential attributes, such as occupation (*i.e.*, employer), education (*i.e.*, college) and location (*i.e.*, home city), is important to many online applications, such as recommendation, personalized search, and targeted advertisement.

Recently, the emergence of social networking websites (*e.g.*, Facebook, Twitter, LinkedIn) brings a new opportunity for user profiling. On these websites, users connect to and communicate with other users. For instance, 1.15 billion users connect to their friends on Facebook, and 283 million users connect to their professional contacts on LinkedIn. Though only a few users voluntarily provide their attributes in their online profiles (*e.g.*, 40% of Facebook users provide their employers [13], and 20% of Twitter users provide their home cities [9]), there is a chance that we can profile the remaining users via their social connections. For example, we can intuitively assume that a user is likely to connect to users who share the same attribute value, and directly apply a majority voting method or its variations to profile a user based on the attribute values observed from his connected friends. In the literature, the early studies on user profiling in social networks [1, 18, 9] are all based on such an assumption.

However, this assumption oversimplifies "noises" of online social networks. For a particular attribute (*e.g.*, employer), a user may connect to many users who have different values from him. For instance, based on our study of 19K social connections on LinkedIn, only 11% of connected users share the same employer, and only 18% of connected users share the same college. Thus, a more realistic assumption is needed to enhance user profiling in social networks.

Insight As our first contribution, we present a novel insight about the *discriminative correlation* between attributes and social connections: *social connections are discriminatively correlated with attributes via a hidden factor – relationship type.* For example, a user's professional contacts on LinkedIn are of implicit relationship types. Some of his contacts are colleagues, who share the same employer but have different colleges; some of his contacts are college classmates, who share the same college but have different employers. This insight clearly captures that, among a user's friends, only some (*e.g.*, colleagues) share the common value with him for an attribute (*e.g.*, employer), and many others do not.

Based on our insight, to accurately profile a user's attribute (*e.g.*, employer), we should propagate attribute values through connections of proper types (*e.g.*, colleagues). However, the relationship types of social connections are implicit, which need to be profiled as well.

Approach As our second contribution, to capture the discriminative correlation insight for user profiling in social networks, we propose a *co-profiling* approach, which jointly

[*] This material is based upon work partially supported by NSF Grant IIS 1018723, the Advanced Digital Science Center of UIUC and the Multimodal Information Access and Synthesis Center at UIUC. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the views of the funding agencies. Chi Wang was supported by a Microsoft Research PhD Fellowship.

WWW'14, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2744-2/14/04.
http://dx.doi.org/10.1145/2566486.2568045.

profiles attributes for users (*e.g.*, employer, college) and relationship types for their connections (*e.g.*, colleagues, college classmates). Such a co-profiling approach not only profiles attributes accurately but also identifies implicit relationship types, which are valuable for many applications, such as grouping a user's friends into different *circles*, and understanding information diffusion within and across circles [20]. To the best of our knowledge, this is the first study that explores the concept of co-profiling.

Specifically, in this paper, we focus on co-profiling upon every user's *ego network*, which means that we profile a user's attributes and the relationship types of his connections given his ego network. An ego network is a network surrounding an individual user, called the ego, which contains connections between the user and his friends and those among his friends. We focus on the ego network of each user instead of a 'full' social network due to several practical reasons. First, a full social network is unavailable to public in general, whereas a user's ego network is often accessible via certain protocols (*e.g.*, user-authorized applications). Second, an ego network keeps important connections related to one user and typically contains hundreds of nodes, which are much more efficient to deal with than a full one with hundreds of millions of nodes.

Algorithm As our third contribution, we develop an efficient algorithm for realizing co-profiling in ego networks based on an optimization framework. In our setting, there are three kinds of variables: user attributes (partially observed, as some friends have known attributes), relationship types (hidden), and connections (observed). To find the hidden variables based on the observed ones, we first conduct a large scale of user studies on LinkedIn to establish the dependencies between attributes and relationship types and between relationship types and connections. Then, we design a cost function to model those dependencies and infer the hidden variables that best satisfy the dependencies given the observed variables (*i.e.*, minimizing the cost). Such a function contains both continuous and discrete variables, which is hard to optimize directly with standard derivatives. We derive an efficient algorithm by alternatively updating each kind of hidden variables based on the idea of the coordinate descent algorithm. Our algorithm captures our insight effectively. Iteratively, it profiles user attributes by propagation through certain types of connections, and profiles relationship types based on inferred attribute values and the network structure.

Evaluation As our forth contribution, we conduct extensive experiments to evaluate our algorithm on a large collection of real-world ego networks. Specifically, we evaluate our proposed algorithm for two tasks, i) *user attribute profiling* and ii) *relationship type profiling.* For the first task, our algorithm profiles ego users' attributes (*i.e.*, employer, college, location) accurately (*e.g.*, 61% accuracy for college), and improves the best baseline significantly (*e.g.*, 12% improvement for college). For the second task, our algorithm correctly identifies relationship types for ego users' connections and improves the best baseline method by 10%.

2. RELATED WORK

As we aim to profile a user's attributes accurately via identifying types of his connections, our work is related to user attribute profiling and relationship type profiling.

User Attribute Profiling has been studied in various scenarios due to their importance in many applications. Most of studies focus on *user-centric* data, such as blog entries [3, 17], query logs [7], and tweets [19, 4, 2] to profile users' certain attributes (*e.g.*, gender and location). For example, Rao *et al.* [19] profile users' gender using a classification method based on features (*e.g.*, words) manually identified from their tweets. Cheng *et al.* [4] profile users' locations based on "local" words (*e.g.*, "Houston") automatically identified from their tweets. However, all these methods do not explore *social connections* for user profiling.

Recently, some studies [13, 1, 18, 9] explore the usefulness of social connections in profiling user attributes. As the seminal study, Mislove *et al.* [13] profile departments of students in a university based on their friendships on Facebook. They assume that students with the same department are likely to be friends and form a community. Thus, they directly apply an existing graph clustering algorithm [5] to find communities, and assign students in a community with the same department. However, such an approach cannot effectively utilize partially available user attributes, and users in a discovered community may not have the same value for any given attribute. Later, Pennacchiotti and Popescu [18] apply heuristics to directly prorogate Twitter users' interests through all their following connections; and Backstrom *et al.* [1] design a probabilistic model to propagate Facebook users' locations via all their friendships.

Our work has two key differences from the above studies. First, we focus on ego-network, while the above methods focus on either a "full" social network [13], which is often impossible to obtain and too large to process, or only the connections between the ego and his friends [1, 18], which ignore useful connections among friends. Second, we capture a new insight that there are different types of connections (*e.g.*, colleagues), each of which only implies certain attributes to be shared (*e.g.*, employer), while the above methods view connections are identical.

User profiling in social network can be viewed as relational classification [10, 11] or graph-based semi-supervised learning [23, 25]. However, those methods generally assume two connected nodes have similar labels, which also cannot capture our insight.

Relationship Type Profiling has been studied only until recently [22, 8, 21, 12], as most studies assume that social connections are homogenous. Leskovec *et al.* [8] employ a logistic regression model to classify positive and negative connections (which imply similar or different interests) in social networks. Tang *et al.* [21] propose a supervised probabilistic model to classify connections into predefined categories (*e.g.*, adviser and advisee). However, as our goal is to profile a user's attributes accurately, our work is to identify implicit circles (relationship types) of friends (connections) (*e.g.*, colleagues) only if they are likely to share some attributes (*e.g.*, employer). We do not classify friends into arbitrary predefined categories, since a predefined category may not align well with an attribute to profile. Recently, McAuley *et al.* [12] build a probabilistic model to group a user's friends into different circles based on their attributes and connections. Our work is different from it, since it assumes that every user has complete attributes, but in our setting only a few users have known attributes.

We note that, while stand graph clustering algorithms [5, 15, 14, 24] can cluster a user's friends into circles, we cannot directly apply them in our setting for two reasons. First, they either do not use users' attributes [5, 15] or assume that all users' attributes are known [14, 24], but only a few users' attributes are known in our setting. Second, they assume that users in a cluster have similar values for all attributes [14, 24], but, as we will see, the friends in a circle only share one or a few attributes (*e.g.*, colleagues share the same employer but have different colleges) in our setting.

Finally, we emphasize that existing methods profile user attribute and relationship types independently, but our work explores the new concept of "co-profiling", which profiles user attributes and relationship types jointly, as they are naturally related in our setting. To the best of our knowledge, this is the first study that explores co-profiling.

3. PROBLEM ABSTRACTION

As motivated in Sec. 1, we aim to study the problem of user profiling in ego networks. In this section, we will formally introduce ego networks and abstract our problem.

First, we formally define users' ego networks. An *ego network* is a (personal) sub network around an individual user, named as the *ego*, in a social network (*e.g.*, Facebook, Linkedin). Generally, it contains the connections between the individual user and his friends and those among his friends. Here, we assume that an ego network is undirected, as many social networks (*e.g.*, Facebook, LinkedIn) are undirected. Our method can be easily extended to handle directed ego networks. As Fig 1 illustrates, a user's ego network can be represented as a graph $G(V, E)$, where vertices V represent users and edges E represent their connections. V contains the ego user v_0 and his friends (*e.g.*, v_1, v_3). E contains both the connections between the ego and his friends (*e.g.*, e_{02}, e_{03}) and the connections among their friends (*e.g.*, e_{13}, e_{23}). In this paper, we specifically use *friends* to refer to the users except the ego user v_0. We use V' to denote the friends, and E' to denote the connections among the friends.

In an ego network, users have attributes (*e.g.*, employer, college, location, ages). We use A_k to denote the k^{th} attribute (*e.g.*, college), and $a_{i,k}$ to denote the value of the k^{th} attribute of a user v_i (*e.g.*, UIUC). To simplify our discussion, we make two assumptions about an attribute. First, we assume that an attribute (*e.g.*, college) has *categorical* values (*e.g.*, UIUC) from a predefined set (*e.g.*, {Stanford, UIUC, Berkeley}), since a numerical attribute (*e.g.*, age) can be categorized (*e.g.*, age-group). Further, we assume that a user has a *single* value (*e.g.*, UIUC) for an attribute (*e.g.*, college). We are aware that, for some attribute (*e.g.*, employer), a user may have multiple values (*e.g.*, v_2 used to work at Google, and now works Facebook now). Our approach can be easily extended for such cases. We will discuss the extension in Sec. 5.

Now, we formally define our problem. In online social networks (*e.g.*, Facebook, Linkedin), many users' attributes are missing, and we aim to profile each user's attributes in his ego networks. Thus, in our setting, we are given a user v_0's ego network $G(V, E)$, where the ego user v_0's attributes are unknown and only some of his friends' attributes (*e.g.*, v_1, v_3) are known, and we need to profile v_0's attributes. We call the friends whose attribute A_k are known as the *labeled* friends for A_k, denoted as L_k, and the friends whose attribute A_k are unknown as *unlabeled* friends for A_k, denoted as U_k. Formally, we abstract our problem as follows.

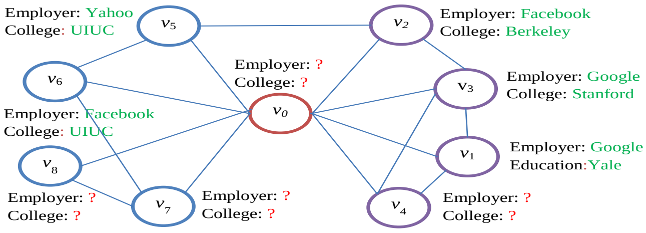


Figure 1: An Example of an Ego Network

User Profiling in Ego Network *Given a user v_0's ego network $G(V, E)$, a list of attributes, A_1, ..., A_m, and some labeled friends $L_k \in V'$ whose attribute A_k are known, we profile the user v_0's attribute value a_k for each attribute A_k, where $k = 1, \ldots, m$.*

We note that the labeled friends L_k could be either identical (*i.e.*, a subset of users who provide complete profiles) or different (*i.e.*, different users provide different attributes in their profiles) for different attributes A_k. Both cases are realistic. For the simplicity of our discussion and notation, we assume the former case and denote the labeled friends as L. Our solution can handle the second case as well.

4. INSIGHT AND APPROACH

To infer the values (*e.g.*, Google) of certain attributes (*e.g.*, employer) of the ego user given the values of those attributes of some of his friends and the connections in the network, we need to properly identify the "correlation" between users' attributes and their social connections, *i.e.*, whether the ego user is likely to share the same value for a given attribute with a friend.

Existing methods [11, 18] often take a simplistic assumption that connections are "identically" correlated with attributes (*i.e.*, all the friends are likely to share the same value with the ego for any attribute). However, this assumption fails to capture that, in an ego network, 1) only a few friends share the same value with the ego for an attribute (*e.g.*, employer), while many friends do not, and 2) for different attributes (*e.g.*, employer and college), the friends who share the same value with the ego are different.

In this paper, we present the *discrimination correlation* insight that social connections are "discriminatively" correlated with attributes (*e.g.*, employer, college, interests) through implicity relationship types (*e.g.*, colleagues, college classmates, club friends). For example, an ego user (*e.g.*, v_0) is more likely to share the employer (*e.g.*, Google) with his colleagues (*e.g.*, v_3, v_2) than with the college classmates (*e.g.*, v_6, v_5), and he is more likely to share interests (*e.g.*, foodball) with club friends than his colleagues or classmates. Such an insight can successfully overcome the limitations of the simplistic assumption used by the exiting methods.

We note that, in an ego network, the ego user's connections (*e.g.*, e_{02}) have a one-to-one mapping to his friends (*e.g.*, v_2). We can group friends to *circles* according to the relationship types of their connections. In the rest of the paper, we will use a circle to refer to a set of friends who have the same type of connections with the ego.

The discriminative correlation insight suggests that, to profile an attribute (*e.g.*, employer) for a user, we should propagate the attribute values from the friends in certain circles (*e.g.*, colleagues v_1, v_3) rather than from all the friends

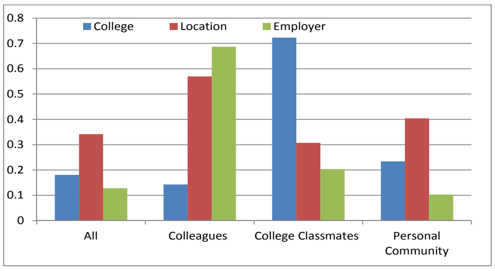


Figure 2: Chance of Sharing Attributes in Circles

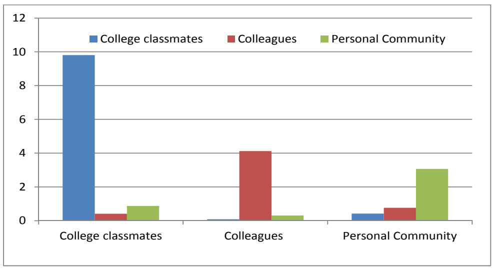


Figure 3: Connections within and across Circles

(*e.g.*, v_6, v_5). However, the circles of friends are latent and also need to be profiled. Since attributes and circles are strongly related (*i.e.*, attributes can help to identify circles, while circles can help to propagate attributes), we take a *co-profiling* approach, which jointly profiles the ego user's missing attributes and latent circles (relationship types, resp.) of his friends (connections, resp.) given an ego network.

We clarify that, as our main goal is to profile the ego's attributes accurately, we target on identifying a circle (*e.g.*, colleagues) only if the friends in it are likely to share some attributes (*e.g.*, employer). We do not intend to classify friends into arbitrary predefined categories (*e.g.*, advisor or advisee) [21] because 1) an arbitrary category may not align well with the attributes to profile, and 2) it is difficult to predefine categories of friends for different attributes in different ego networks (*e.g.*, student users share their locations with their classmates, while employee users share their locations with their colleagues).

We are aware that some attribute (*e.g.*, name, gender) is associated with no circle or all circles. However, as we discussed above, for many important attributes (*e.g.*, employer, college, locations, interests), our key insight is valid. We focus on profiling these circle-related attributes. Further, as we will discuss in Sec. 5, our approach can profile circle-independent attributes as good as existing social network based user profiling methods.

5. CO-PROFILING ALGORITHM

In this section, we realize "co-profiling" in an optimization framework. In our setting, there are three kinds of information: attributes (partially known), circles (unknown), and connections (known). To infer the unknown information based on the known information, we first investigate their dependencies following our insight. Then, we propose a cost function to model those dependencies, so we can infer the unknown information that best satisfies the dependencies, *i.e.*, minimizing the cost. Finally, we derive an efficient algorithm to solve the optimization problem.

5.1 Dependencies

To enable co-profiling, we need to establish dependencies between known and unknown information. Our insight tells that attributes are discriminatively correlated with connections via latent circles. Following our insight, we investigate a large collection of real-world ego networks to characterize 1) the dependency between attributes and circles and 2) the dependency between circles and connections. Specifically, we conduct a user study to collect ego networks from LinkedIn, and obtain circles (relationship types) of ego users' friends (connections). Sec. 6 describes our study and the dataset in detail. Via investigating our collected dataset, we obtain two important observations.

- **Attribute-Circle Dependency:** *the friends in a circle share the same value with the ego user for certain attributes.* Fig. 2 shows the percentages of friends who have the same attribute value with the ego for three attributes (*i.e.*, employer, college, location) in different circles (*i.e.*, colleagues, college classmates, personal community) and in general. In a given circle (*e.g.*, colleagues), the friends are much more likely to share the same value for some attribute (*e.g.*, college) with the ego than those in other circles (*e.g.*, college classmates) or in general. Here, we emphasize that friends in a circle (*e.g.*, college classmates) only share one or a few attributes (*e.g.*, college) instead of all attributes.
- **Circle-Connection Dependency:** *friends across circles are loosely connected.* Fig. 3 shows the average connections per user within and across three different circles (*i.e.*, colleagues, college classmates, personal community). The average number of connections for each user within the same circle (*e.g.*, ten connections per user within college classmates) is significantly higher than the average number of connections across circles (*e.g.*, less than one connection per user from college classmates to colleagues).

As a noteworthy fact, for attributes such as employer and college, friends in general have different values from the ego (*e.g.*, only about 11% of all the friends have the same employer). It clearly illustrates the challenge of user profiling.

5.2 Model

Now, we discuss how to formally model the above dependencies in an optimization framework. Intuitively, we can assume that there are K circles in a given ego network, and the friends in each circle share the same value (*e.g.*, UIUC) for certain attributes (*e.g.*, college) with the ego user according to the first dependency and have few connections to the friends in other circles (*e.g.*, colleagues, personal community) according to the second dependency.

We emphasize that K should be a variable to be learned. First, multiple circles may share a common attribute or a circle may share multiple attributes, so K may not be equal to the number of attributes. Second, different ego networks have different numbers of circles. We discuss how to determine K in Sec. 5.4, and let us assume K is fixed for now.

It is also worthwhile to point out that we cannot simply apply standard methods for graph clustering with attributes [24, 14, 12] for two reasons. First, they assume that node attributes are complete, but attributes of many users are unknown in our setting. Second, they assume that nodes in a cluster are similar across all attributes, but friends in a circle only share the same value for one or a few attributes (*e.g.*, college classmates share the same college but have different employers) in our setting.

To design our objective function, we first introduce our mathematic abstraction of attributes and circles. As the first dependency suggests that the friends in a circle are likely to share the values with the ego for one or a few attributes, we also give our mathematic abstraction for the association between attributes and circles.

- To represent attribute values of a user v_i, we introduce an *attribute vector* f_i to represent all the attributes (*e.g.*, employer, college) of v_i. Each dimension of f_i represents a candidate value of certain attribute (*e.g.*, Google for employer or UIUC of college). The candidate values (*e.g.*, Google, Facebook) for an attribute (*e.g.*, employer) can be obtained from the labeled friends or given by a predefined dictionary. The value of the y^{th} dimension of f_i, denoted as $f_{i,y}$, is a real number from 0 to 1, which indicates how likely v_i is associated with the attribute value represented by the y^{th} dimension. For labeled friends $v_i \in L$, we assign $f_{i,y}$ to 1, if the corresponding attribute value is observed in his profile, and 0, otherwise. For any unlabeled friend $v_i \in U$ or the ego user v_0, f_i is an unknown vector. Further, some circle (*e.g.*, club friends) might be formed based on an unspecified value (*e.g.*, IBM) of a given attribute (*e.g.*, employer) or the values of other attributes (*e.g.*, interests) besides the given attributes (*e.g.*, employer, college). Thus, we add an additional "catchall" dimension in every user's attribute vector and assign a small score (*e.g.*, 0.1) to allow circles to be formed by the catchall dimension. For example, given the college attribute, which has three candidate values (*e.g.*, 1 Berkeley, 2 UIUC, 3 Stanford), and the employer attribute, which has two candidate values (*e.g.*, 4 Google, 5 Faceook), the attribute vector of a user who studies at Berkeley and works for Google is $< 1, 0, 0, 1, 0, 0.1 >$.
- To represent circles of friends, we introduce a latent variable *circle assignment* x_i for each friend $v_i \in V'$. x_i can be any value from 1 to K, which indicates the circle that v_i belongs to. Thus, a circle C_t can be defined as $\{v_i \in V' | x_i = t\}$. We note that, the ego user connects (or belongs) to all the circles in his ego network (*e.g.*, colleagues in Google, college classmates in UIUC), so we do not need the circle assignment for v_0. For simplicity we do not model the circle overlapping and it can be considered as future work.
- To represent the associations between circles and attributes, we introduce an *association vector* w_t for each circle C_t. The y^{th} dimension of w_t, denoted as $w_{t,y}$, is a binary value, which indicates whether C_t is associated with the attribute value represented by the y^{th} dimension in the attribute vector f_i (*i.e.*, whether the friends in C_t share the corresponding attribute value with the ego). Given the same example mentioned above, the association vector $< 0, 1, 0, 0, 0, 0 >$ of a circle represents that the friends in this circle study in UIUC with the ego user and have different values for other attributes. Since we do not know which attribute values are associated with a circle C_t, w_t is an unknown vector. As the first dependence tells, friends in a circle only share one or a few attribute values with the ego, w_t should have only one or a few dimensions to be non-zero. To simplify our discussion, let us assume w_i has one dimension equal to one, since, in most cases, the friends in a circle share one attribute value with the ego (*e.g.*, college classmates only share the same college). We discuss how to extend this assumption in Sec. 5.4.

Now, we model the attribute-circle dependency based on the above abstraction. The dependency suggests the friends in a circle C_t (in other words, the friends who connect to the ego through the connections of a latent relationship type t) share the attribute value designated by w_t with the ego user v_0. Further, since two connected friends in C_t also have the same relationship type t, they should also share the attribute value designated by w_t. Thus, for every two connected users v_i and v_j in $C_t \cup \{v_0\}$, their attribute vector f_i and f_j should be close on the dimension designated by w_t. Using a standard squared distance function, we tend to minimize $\sum_{v_i \in C_t}(w_t \cdot (f_0 - f_i))^2 + \sum_{e_{ij} \in E', v_i, v_j \in C_t}(w_t \cdot (f_i - f_j))^2$. Further, the labeled friends L provide explicit knowledge for discovering the associated attribute value of a circle. Specifically, our model should select the associated attribute value for a circle C_t that is shared by many labeled friends in C_t, which is to minimize $\sum_{v_i \in L \cap C_t}(w_t \cdot f_i - 1)^2$.

Then, we model the circle-connection dependency. It suggests that the friends in a circle do not have many connections to the friends in other circles. Thus, our model should minimize the connections cross different circles, which is $\sum_{e_{ij} \in E', x_i != x_j} 1$. This term is the *cut* used by many graph clustering algorithms. Here, we use it to capture of the second dependency. However, as it is only a part of our overall cost function, our model is different from K-cut graph clustering. We note that other graph clustering objectives (*e.g.*, normalized cut or modularity) can also be applied, and we choose cut because circles may not necessarily be balanced.

Finally, we linearly combine the three terms together in our cost function, which is Eq. 1.

$$\lambda_1 \sum_{t=1}^{K} \{ (\sum_{e_{ij} \in E', v_i, v_j \in C_t} (w_t \cdot (f_i - f_j))^2 + \sum_{v_i \in C_t} (w_t \cdot (f_0 - f_i))^2) \} + \lambda_2 \sum_{t=1}^{K} \sum_{v_i \in L \cap C_t} (w_t \cdot f_i - 1)^2 + \lambda_3 \sum_{e_{ij} \in E', x_i != x_j} 1 \quad (1)$$

Here, as the weights are free to scale, we confine the weight for the third term λ_3 as one. We will discuss how to set the parameters λ_1 and λ_2 in our algorithm. We are aware that, beside our objective function, there might be other ways to model the two dependencies (*e.g.*, probabilistic framework). We plan to explore them as our future work.

5.3 Algorithm

Now, we derive our co-profiling algorithm. Specifically, we want to design an efficient algorithm, which finds the solutions for unknown variables that jointly minimize Eq. 1. There are three groups of unknown variables, circle assignments x_i for each friend $v_i \in V'$, the attribute vector f_i for each unlabeled friend $v_i \in U$ and the ego user v_0, and the association vector w_t for each circle C_t. Such an objective function is hard to minimize, as it involves not only continuous variables (*e.g.*, f_i) but also discrete variables (*e.g.*, x_i), for which we cannot do derivatives. Based on the idea of the coordinate descent algorithm, we derive our algorithm by iteratively updating unknown variables. Specifically, we discuss how to update each group of variables.

First, we focus on how to update the attribute vector f_i for each unlabeled friend $v_i \in U$ and the ego user v_0, when the circle assignment x_i of each friend $v_i \in V'$ and the association vector w_t for each circle C_t are fixed. The function becomes

$$\sum_{t=1}^{K}((\sum_{e_{ij} \in E', x_i, x_j \in C_t} (w_t \cdot (f_i - f_j))^2 + \sum_{v_i \in C_t} (w_t \cdot (f_i - f_0))^2)$$

which is a quadratic function and can be solved in polynomial time with a standard coordinate descent algorithm. Specifically, we iteratively apply the following update rules until the above function converges. The rules are derived based on finding the first order partial derivative for each unknown variable in the above function, and have the convergence and optimality guarantees. Since w_t has only one nonzero dimension, which we denote as $w_{t,y}$, we only use Eq. 2 to update $f_{i,y}$ of f_i, and do not update other zero dimensions. Similarly, we use Eq. 3 to update f_0 for any non-zero dimensions selected by all the circles and will not update f_0 on other dimensions.

$$f_{i,y} = \frac{f_{0,y} + \sum_{e_{ij} \in E', v_j \in C_t} f_{j,y}}{1 + \sum_{e_{ij} \in E', n_j \in C_t} 1}, v_i \in U \cap C_t, w_{t,y} = 1 \quad (2)$$

$$f_{0,y} = \frac{\sum_{t=1, w_{t,y}=1}^{K} \sum_{v_j \in C_t} f_{j,y}}{\sum_{t=1, w_{t,y}=1}^{K} \sum_{v_j \in C_t} 1}, w_{t,y} = 1, \forall t = 1, ...K. \quad (3)$$

The update functions can be viewed as label propagation from neighbors. While they look similar to graph-based semi-supervised learning [23] or relational classification [11], there are two key differences. First, our functions are circle-aware, which propagate attributes from neighbors only in the same circle rather than all the neighbors. Second, our functions are attribute-aware, which only propagate the associated attribute value of the circle instead of all attributes. Thus, our algorithm captures our key insight and is robust to noises in online social networks.

Second, we discuss how to update the circle assignment x_i for each friend $v_i \in V'$, when the attribute vector f_i for each user $v_i \in V$ and the association vector w_t for each circle C_t are fixed. This can be shown to be an NP-hard problem. Like other clustering algorithms [5, 15], we develop a heuristic algorithm to find a suboptimal solution for the problem. Intuitively, our algorithm iteratively revises a user v_i's circle assignment x_i to minimize the objective function. Specifically, in every iteration, we use the following equations to decide a friend's circle assignment x_i greedily such that it reduces the value of Eq. 1 to the largest degree (or stay unchanged if no other assignment can reduce it).

$$\begin{aligned} x_i &= \arg\max_{t=1,\ldots,K} \Big[\sum_{e_{ij} \in E', v_j \in C_t} \big(1 - \lambda_1 (w_t \cdot (f_i - f_j))^2\big) \\ &\quad -\lambda_1 (w_t \cdot (f_i - f_0))^2 \Big], v_i \in U \qquad (4) \end{aligned}$$

$$\begin{aligned} x_i &= \arg\max_{t=1,\ldots,k} \Big[\sum_{e_{ij} \in E', v_j \in C_t} \big(1 - \lambda_1 (w_t \cdot (f_i - f_j))^2\big) \\ &\quad -\lambda_1 (w_t \cdot (f_i - f_0))^2 - \lambda_2 (w_t \cdot f_i - 1)^2 \Big], v_i \in L \quad (5) \end{aligned}$$

These update functions can be intuitively explained. Specifically, Eq. 4 finds a circle C_t of v_i so that v_i has many connections within C_t. The weight of a connection is adjusted according to the difference of the two connected users on the associated attribute value (*i.e.*, $(w_t \cdot (f_i - f_j))^2$). The weight is reduced, if two connected users have different values. λ_1 weights how much the weight of a connection should be reduced according to the difference. Since a connection to a circle should be a positive evidence of belonging to that circle, λ_1 should be less than one. Further, Eq. 5 shows that, for a labeled friend v_i, if v_i has the attribute value associated with a circle, v_i should be in this circle. λ_2 weighs the importance of matching on an attribute. As we aim to find attribute-related circles, λ_2 should be large. It is clear that our algorithm is different from clustering a network based only on connections [5, 15]. While the algorithm does not guarantee that we find the optimal circle assignments, it improves the solution at every iteration and runs efficiently.

Third, we discuss how to update the association vector w_t for each circle C_t, when each user v_i's attribute vector f_i and the circle assignment x_i are fixed. With the assumption that w_t has only one dimension equal to one, we can exhaustively enumerate all the candidates and select the best w_t. Thus, for each circle, we only need to enumerate a limited number of candidates, which is equal to the number of dimensions of the attribute vector. Further, w_t is independent of each other when x_i and f_i are fixed, so we can decide w_t for each circle independently.

As a whole, our algorithm conducts the above three steps iteratively until it converges. Intuitively, at the first step, it utilizes circles to accurately propagate certain attributes; at the second step, it utilizes the propagated attributes and the network structure to form circles; and at the third step, it automatically finds the associated attribute value for each circle. The algorithm naturally combines attribute prorogation and network clustering with a principled optimization approach. Such an algorithm realizes "co-profiling", which is new and has not been discovered in the literature.

Finally, for an attribute, we select the candidate attribute value with the maximum value in the ego's attribute vector f_0 as the true value for the attribute. We note that, to compare values propagated from different circles and avoid bias to small circles, we compute a final f_0 in the end of iterations and do not do the normalization in Eq. 3.

Convergence We can show the convergence of our algorithm. Intuitively, at each step, our algorithm tries to improve the cost function and makes sure that the cost function is non-decreasing. Further, as there is a minimum value for the cost function (*i.e.*, 0), our algorithm converges. We are aware that it might converge to a local optimum, and our results may be affected by different initialization methods. However, as our experiments show, our algorithm works well in practice with reasonable/standard initialization. Specifically, we initialize each dimension of f_i for an unlabeled friend as 0.5, use a standard clustering algorithm [5] to initialize the circle assignments, and then start to update variables iteratively.

Complexity The complexity of our algorithm is polynomial to the number of the nodes in an ego network, which is very efficient. Specifically, in an iteration, the first step takes $O(|E|D)$ to update attribute vectors, where D is the number of dimensions of an attribute vector (most attributes contain less than 10 candidate values for an ego), the second step takes $O(|V|K)$ to update assignments iteratively, and the third step enumerates at most $O(DK)$ candidates.

5.4 Discussion

Further, we give some discussions about our algorithm.

First, we discuss how to determine the number of circles K. Like many other clustering algorithms [12, 5, 16], K is an important parameter to learn. We can use a standard technique [12, 5], which selects K by trying different values and selecting the best one according to an objective function. Since an ego network is small and our algorithm runs efficiently, it is possible for us to test different K. To choose a proper K initially, we run a standard clustering algorithm first [5] to select an initial value.

Second, we discuss how to extend our algorithm to handle multiple-valued attributes. First, when we initialize f_i, if one of v_i's attribute values is equal to the value represented by the y^{th} dimension, we initialize $f_{i,y} = 1$. Second, when we profile the ego user, we can take the top n values from f_0 as its values or use the threshold to determine whether a value should be true or not.

Third, to handle the cases that a circle may share more than one attributes, we could relax the association vector to have multiple nonzero dimensions. A heuristic but fast way is to assign several dimensions of w_t to one together, when each individual dimension achieves similar costs for circle C_t's portion in Eq. 1, and then normalize it. We note that our algorithm naturally handles the cases where some circles are associated with a common attribute.

Finally, we explain that our algorithm can handel the attributes that do not induce any circles or induce many circles as good as existing methods (*e.g.*, majority voting). Specifically, for the attributes that do not induce any circles, our algorithm will associate circles to other attributes (if any) or the catchall dimension, which we add to the user attribute vector to generally represent extra attributes. For this case, we can directly run a backup method (*i.e.*, majority voting) to profile users' attribute based on his friends. For the attributes that may induce many circles, our algorithm will take majority voting from the friends in those circles, which works like the majority voting based method.

6. EXPERIMENTS

In this section, we conduct extensive experiments to evaluate our co-profiling (CP) algorithm. Given a user's ego network, CP not only profiles his attributes but also identifies relationship types (circles) of his connections (friends). Thus, we evaluate CP for two tasks, 1) *attribute* profiling (AP) and 2) *relationship type* (circle) profiling (RP).

6.1 Dataset

To enable evaluation, we collect a large set of real-world ego networks from LinkedIn during 2013.

We first survey several publicly available social network datasets [12, 6], and find them not suitable for our experiments. McAuley and Leskovec [12] collect users' ego networks from Facebook and Google+. However, users' attribute values are tokenized and anonymized as binary vectors, so we cannot use it for AP. Gong *et al.* [6] collect a sub-network in Google+ with users' attributes. However, the types (circles) of relationships (friends) are unknown, so we cannot use it for RP.

Thus, we collect a new dataset from LinkedIn for our experiments. We choose LinkedIn, because users on LinkedIn are likely to provide important attributes such as employer and college. To collect our data, we send invitations to a list of users (*e.g.*, graduate students and professors in several universities) to invite them to an online study, and then those users can further invite their friends to join the study. If a user participates in our study, we collect his attributes (*e.g.*, *employer*, *college*, and *location*) and his ego network (including the connections from the ego user to his friends and those among his friends) based on Linkedin APIs. We also ask the ego user to label the relationship types of his friends from several categories (*e.g.*, *college classmates*, *colleagues* and *personal communities*). About 268 users participate in our study and label their friends. Due to Linkedin APIs' limitations and users' privacy settings, 175 ego networks are successfully collected. These ego users are associated with 193 different *colleges*, 375 different *employers*, and 52 different *locations*, and their ego networks contain about 19K users and 110K connections, among which 8K are labeled.

6.2 Attribute Profiling

Since our main goal is to profile user attributes, we first evaluate our algorithm for profiling ego users' attributes.

6.2.1 Experiment Configuration

We set up the task to profile three different attributes, including *employer*, *college*, and *location*. We choose these attributes, because they are important for many applications and we can collect their ground truth from users' LinkedIn profiles. For each ego user in our dataset, we randomly sample a small percentage (*e.g.*, 20%) of his friends as labeled friends (whose attributes are revealed to our algorithm and baseline methods) with a uniform sampling method, and then we profile the ego's attributes based on the attributes of the labeled friends and the ego network structure with our algorithm and baseline methods.

We use the *accuracy* to measure the performance of a method. It is the ratio of the number of the correctly profiled users to the total number of test users. Since, for some attribute (*e.g.*, employer), a user may have multiple values (*e.g.*, Google, Facebook), we define that a user is correctly profiled if the profiled value matches any of his true values.

To show the effectiveness of our co-profiling algorithm, we compare it with three baseline methods.

- AP$_w$: As we can view AP as a relational classification problem (*i.e.*, classifying a node based on its neighbors), we use a simple but effective relational classifier, the *relation neighbor classifier* [11], as our first baseline. This method has also been applied to AP in previous studies [18]. Specifically, AP$_w$ is based on the assumption that two connected nodes are likely to share the same attribute value, and calculates the probability that a user associates with an attribute value as the weighted average of the probabilities of its labeled friends (*i.e.*, weighted majority voting). In our experiments, we try different weighting schemes, such as uniform weights and the number of shared friends, and use the one with the best performance. We emphasize that, while this method is simple, as previous studies show [11, 18], it performs surprisingly well in many settings, even compared to complex models.
- AP$_i$: As we can view AP as a graph-based semi-supervised learning (GSSL) problem [25, 23] (*i.e.*, classify nodes in

a graph), we use a widely applied GSSL method [23] as our second baseline. In the GSSL setting, the graph needs to be constructed based on nodes' features first. In our setting, the graph (*i.e.*, the ego network) is given. AP$_i$ *iteratively* estimates every user's attribute value according to all its neighbors (including both labeled and unlabeled ones) and its initial value (if any). It advances AP$_w$, because it utilizes additional connections 1) between the ego user and his unlabeled friends and 2) among his friends to propagate attribute values iteratively. We note that AP$_i$ can be also viewed as a collective inference method in relational classification, which applies the relation neighbor classifier for every node at each iteration.

- AP$_c$: We also use a method [13] recently proposed for user profiling in a full social network as our third baseline. Specifically, it starts from labeled users with the same attribute value, applies an existing local clustering algorithm based on [5] to find new users, and assigns those users with the same value. We note that AP$_c$ is proposed for user profiling in a full social network. Here, we adapt it to each user's ego network. We can view AP$_c$ as a baseline, which first applies an existing clustering algorithm to find circles in an ego network and then profiles the ego user based on discovered circles.

Note that, to fairly compare CP with these baselines, which profile each attribute independently, CP profiles one attribute at a time and does not take multiple attributes as input.

6.2.2 Experiment Results

In our experiments, we first compare CP with the three baselines, and then investigate CP specifically.

First, we compare CP with the three baselines for profiling *employer*. Here, for each test ego user, we randomly select 20% of his friends as labeled friends, whose attributes are revealed. To make sure our results are statistically sound, we repeat our experiments five times with different samples, and report the mean and the variance of accuracies in five rounds. Fig. 4 shows the results, from which we can obtain the following observations.

- AP$_w$ achieves a reasonable accuracy (*i.e.*, 53%) when using 20% of friends as labeled friends, which shows that social connections are useful for profiling user attributes. However, it performs the worst among the four methods. The reasons could be 1) it fails to utilize additional connections available in ego networks and 2) the assumption that two connected users are likely to share the same attribute does not precisely capture the correlation between users' attributes and their social connections.
- AP$_i$ further improves AP$_w$, since AP$_i$ utilizes additional connections to propagate attribute values iteratively, which shows that additional connections are helpful. However, AP$_i$ performs much worse than CP because AP$_i$ relies on the same assumption used in AP$_w$. It neither precisely captures the correlation between users' attributes and their social connections.
- AP$_c$ performs only slightly better than AP$_w$, and worse than AP$_i$, because it directly applies an existing clustering algorithm, which cannot effectively utilize available attributes from the labeled friends to find circles, and the users in such a discovered circle may not share the same attribute value.

Accuracy of Profiling Employer				
Method	AP$_w$	AP$_i$	AP$_c$	CP
Mean	0.53	0.56	0.54	**0.60**
Variance	2E-03	2E-04	4E-04	4E-05
Accuracy of Profiling College				
Method	AP$_w$	AP$_i$	AP$_c$	CP$_g$
Accuracy	0.50	0.54	0.50	**0.61**
Variance	9E-05	1E-04	8E-04	2E-05
Accuracy of Profiling Location				
Method	AP$_w$	AP$_i$	AP$_c$	CP$_g$
Accuracy	0.62	0.61	0.60	**0.68**
Variance	9E-04	8E-05	9E-04	5E-04

Figure 4: Attribute Profiling Results

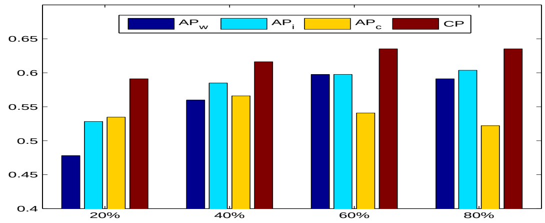


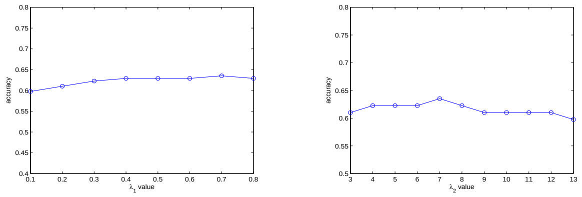
Figure 5: Accuracy with Different Percentages of Labeled Friends

- CP performs the best. It clearly suggests that 1) our insight can better model the correlation between users' attributes and their social connections than the simple assumption used in the baselines, and 2) our algorithm, which iteratively infers user attributes and identifies types of connections, captures our insight effectively.

We note that the variances here are small, which suggest that our results are reliable. We further conduct a significant test between our method and the baseline methods, and our method successfully passes the significant test with the p-value equal to 0.05.

Second, we show the results of profiling *location* and *college* in Fig. 4. For both attributes, our method achieves high accuracies and performs better than the baselines. The results clearly suggest that our method is generally effective for profiling attributes. Here, we make two notes. First, AP$_c$ performs even worse than AP$_w$, which again suggests that it cannot effectively utilize available attributes from the labeled friends. Second, while CP improves AP$_w$, AP$_i$ does not improve AP$_w$ for profiling location. It suggests that AP$_i$ does not utilize the connections effectively, since it directly propagates attributes regardless of the latent connection types. Due to space limits, we will focus on the employer attribute in the rest of experiments.

Third, we compare CP with baselines when different percentages of friends' attributes are known. Fig. 5 shows the accuracies of different methods for profiling employer using different percentages of friends as labeled friends. We obtain the following observations. First, the performance of a method increases as the number of labeled friends increases for all the methods except AP$_c$, which suggests additional attributes are helpful. Here, AP$_c$ is an exception, since, as we mentioned, it cannot effectively utilize available attributes. Second, our algorithm consistently outperforms the baseline methods with different percentages of friends as labeled friends, which again demonstrates the effective-



(a) $\lambda_1 = 0.1 - 0.8$, $\lambda_2 = 3$ (b) $\lambda_1 = 0.7$, $\lambda_2 = 1 - 13$

Figure 6: CP **with Different Parameter Values**

ness of our algorithm, especially the ability of profiling with low percentage of available attributes.

The above experiments clearly demonstrate that our algorithm advances the existing methods for attribute profiling.

Finally, we investigate the impact of parameters in our algorithm. Recall that there are two parameters, λ_1 and λ_2. As Eq. 5 suggests, λ_1 is to adjust the weight of a connection according to the difference between the two connected users, and λ_2 is to weigh the importance of labeled friends' attributes. In our experiments, we obtain their values with a standard parameter tuning procedure based on a small validation set. Fig. 6 shows the accuracies of CP with different λ_1 and λ_2 respectively. We have two observations. First, while the parameters impact the performance, the performance varies a little when the parameters are in reasonable ranges. Second, those ranges can be reasonably explained. Particularly, the optimal λ_1 is around 0.7, as we should reduce the weight of a connection when two attribute vectors have some differences. The optimal λ_2 is around 7, which suggests that the labeled friends' attributes are important for identifying attribute-related circles.

6.3 Relationship Type (Circle) Profiling

As our co-profiling algorithm can identify latent relationship types (circles) of a user's connections (friends), we also evaluate its performance for relationship type profiling (RP).

6.3.1 Experiment Configuration

We set up the task to identify two attribute-related circles (relationship types) for each given ego network. They are *colleagues*, in which the friends are likely to share the same employer with the ego user, and *college classmates*, in which the friends are likely to share the same *college* with the ego user. We use the same experiment setting as the previous task, *i.e.*, in a given ego network, there are only a few friends whose attributes are known. We emphasize that circle assignments (relationship types) for friends (connections) are only used for evaluation.

We evaluate our algorithm and other baseline methods on our dataset described in Sec. 6.1. Specifically, it contains about 8K *labeled* friends (connections) from the collected ego networks. Among them, about 500 friends are labeled as colleagues, and about 2600 friends are labeled as college classmates. All the labels are given by the ego users, so we can trust these labels.

To measure the performance of a method, we use *precision*, *recall* and *F1*. They are standard metrics used in classification and information retrieval. In this task, precision is the number of friends correctly profiled as belonging to a circle (*e.g.*, colleagues) divided by the total number of friends profiled as belonging to the circle, recall is the number of friends correctly profiled as belonging to a circle divided by the total number of friends in the circle, and F1 is the harmonic mean of precision and recall.

Method	RP$_a$	RP$_n$	RP$_{an}$	CP
Recall	0.13	0.39	0.43	**0.45**
Precision	**0.90**	0.64	0.51	0.69
F1	0.22	0.48	0.46	**0.54**

Figure 7: Results for Profiling Colleagues

Method	RP$_a$	RP$_n$	RP$_{an}$	CP
Recall	0.09	0.39	0.43	**0.49**
Precision	**0.98**	0.95	0.94	0.97
F1	0.16	0.55	0.59	**0.65**

Figure 8: Results for Profiling College Classmates

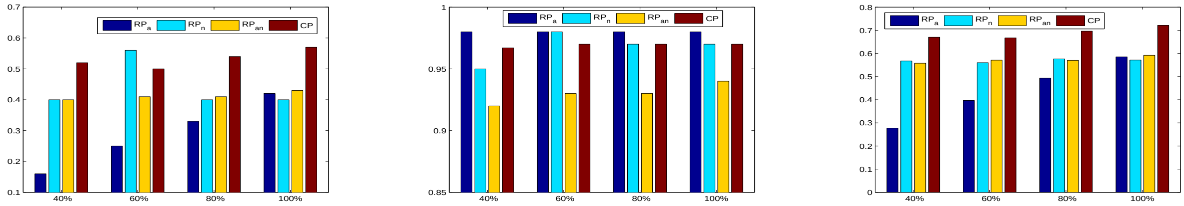
To demonstrate the effectiveness of our algorithm CP, we compare it with three baseline methods.

- RP$_a$ profiles circles in an ego network based on users' attributes. Intuitively, a friend is likely to belong to a target circle (*e.g.*, colleagues), if he shares the same value with the ego user for the related attribute (*e.g.*, employer). However, the ego user's attributes are unknown in our setting. Thus, to profile circles based on attributes, RP$_a$ first uses an attribute profiling method AP$_w$ to profile the ego's attributes based on his labeled friends. We use RP$_a$ as our first baseline to examine how well we can profile circles based only on given and profiled attributes.
- RP$_n$ profiles circles in an ego network based on the network structure [5]. Intuitively, friends in a circle (*e.g.*, colleagues) are likely to form a cluster in an ego-network. Thus, we use a modularity based graph clustering method [5] as our second baseline. We choose this method because it is widely applied in many scenarios (*e.g.*, attribute profiling task [13]) and can automatically select K for a given network. As the algorithm only finds K clusters, RP$_n$ further uses attributes of labeled friends in each cluster to identify the circles that we are interested in. Intuitively, RP$_n$ selects a cluster, in which the friends are likely to share the related attribute (*e.g.*, employer) of the circle of interest (*e.g.*, colleagues). We compare our algorithm with RP$_n$ to show how much we can improve the methods that profile circles based only on the network structure.
- RP$_{an}$ profiles circles in an ego network based on both users' attributes and the network structure [12], which is a state of the art method for discovering circles in ego networks. However, it assumes that every user has a complete profile, which may fail to handle the challenge that only a few users provide their attributes. We compare with this baseline to assess the advantage of our algorithm when dealing with partially available attributes. This method also only outputs K circles, and we apply the strategy used in the previous baseline to identify the two circles (*i.e.*, colleagues and college classmates).

6.3.2 Results

Now, we present our experiment results for this task. Here, we focus on comparing our algorithm CP with the baselines.

Fig. 7 shows the results for profiling *colleagues* in terms of precision, recall, and F1. Here, for each ego user, we randomly select 20% of his friends as labeled friends whose employers are revealed like in the first task. We can obtain the following observations.



(a) Recall

(b) Precision

(c) F1

Figure 9: Results for Profiling College Classmates with Different Percentages of Labeled Friends

- RP$_a$ performs the worst (the lowest F1) among the four methods. Specifically, it has a high precision but a very low recall. The precision is high, indicating that user attributes (*e.g.*, employer) are reliable signals for profiling attribute-related circles (*e.g.*, colleague). The recall is low, because only a few friends have known attributes and RP$_a$ can only make a limited number of predictions. It suggests that, in online social networks, identifying circles with only known attributes is far from enough.
- RP$_n$ performs well in all the three measures. It indicates that the network structure is useful for profiling circles. However, it performs worse than CP in all the three measures, which suggests that the network structure is also insufficient for identifying circles.
- RP$_{an}$ performs similarly as (slightly worse than) RP$_n$ in terms of F1, although it utilizes both user attributes and the network structure. There are two reasons. First, RP$_{an}$ assumes every user has a complete profile, which contains values of different attributes, while values of only one attribute from a few friends are given in our setting. Second, it also does not capture the sparse association between attributes and circles (*i.e.*, one circle is associated with only a few attributes). The results here suggest that available attributes should be carefully used when finding attribute-related circles.
- CP achieves the best F1. It greatly improves RP$_a$ and RP$_n$ because it utilizes both the network structure and users' attributes. We note that, though CP's precision is lower than RP$_a$ (0.90 v.s. 0.70), CP's recall is much higher than RP$_a$ (0.45 v.s. 0.13). CP also greatly improves RP$_{an}$, because 1) it is designed to utilize partially available attributes and 2) it models the sparse dependency between attributes and circles properly.

Figure 8 shows the results for profiling *college classmates*. Generally, we can obtain similar findings as the previous experiment. For example, CP significantly improves the three baselines. The results show that our approach is general to identify different attribute-related circles.

Further, we investigate the performance of each method using different percentages of friends as labeled friends. Fig. 9 shows the precision, recall, and F1 for identifying college classmates. We obtain the following observations.

- RP$_a$: Its recall and F1 increase as the number of the labeled friends increases, since RP$_a$ solely relies on user attributes to profile. Especially, when many friends' attribute values are known, RP$_a$ performs reasonably well. The results again validate that user attributes are strong evidence for identifying attribute-related circles (*e.g.*, colleagues) and should be utilized. However, we should be aware that it is unlikely to have all users' attribute values in many social networks.
- RP$_n$: It benefits from additional labeled friends slightly, because it mainly utilizes the network structure to group friends (form circles) and only uses friends' attributes to identify the target circle. It performs better than RP$_a$ in most cases, which again confirms the usefulness of the network structure for this task.
- RP$_{an}$: Its performance also sightly increases as the number of the labeled friends increases, since it also utilizes attributes to identify circles. However, as it assumes that all attributes (*e.g.*, employer, college, interests) for every user are known, its performance is not as good as CP.
- CP: Its performance increases as the number of labeled friends increases, which suggests that our approach can make good use of available attributes. Further, our algorithm performs the best in all the cases, which clearly demonstrates our approach is effective in identifying attribute-related circles.

Based on the above experiments, we can conclude that our algorithm can correctly find attribute-related circles (*e.g.*, colleagues), and thus it can profile users' attributes based on corresponding circles accurately.

7. CONCLUSION

In this paper, we study the problem of profiling users' attributes based on their ego networks. We make the following contributions to the problem. First, we present a new insight to capture the correlation between attributes and social connections. Second, we propose a novel *co-profiling* approach to profile users' attributes and circles of their friends jointly. Third, we develop an efficient algorithm, which profiles user attributes by propagation from friends in certain circles, and profiles circles for friends based on inferred attribute values and the network structure. Forth, we conduct extensive experiments based on real-world ego networks to evaluate our algorithm. The results show that our co-profiling algorithm not only profiles users' attributes accurately, improving the state-of-the-art methods greatly, but also correctly identifies latent circles of users' friends, which are useful for many advanced applications.

As our future work, we would like to evaluate our algorithm in a large dataset and apply it to different social networks (*e.g.*, Twitter and Facebook). Further, we want to extend our algorithm with refined circle assumptions. For example, we may explore the overlapped circles, where a friend may belong to multiple circles, or hierarchical circles to profile hierarchical attributes. We are also interested in exploring our co-profiling approach in a probabilistic framework.

8. REFERENCES

[1] L. Backstrom, E. Sun, and C. Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *WWW '10*, pages 61–70, 2010.

[2] J. D. Burger, J. Henderson, G. Kim, and G. Zarrella. Discriminating gender on twitter. In *EMNLP '11*, pages 1301–1309, 2011.

[3] J. D. Burger and J. C. Henderson. An exploration of observable features related to blogger age. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 15–20, 2006.

[4] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *CIKM '10*, pages 759–768, 2010.

[5] A. Clauset, M. Newman, and C. Moore. Finding community structure in very large networks. *Physical review E*, 70(6), 2004.

[6] N. Z. Gong, W. Xu, L. Huang, P. Mittal, E. Stefanov, V. Sekar, and D. Song. Evolution of social-attribute networks: Measurements, modeling, and implications using google+. In *IMC '12*, pages 131–144, 2012.

[7] R. Jones, R. Kumar, B. Pang, and A. Tomkins. "i know what you did last summer": query logs and user privacy. In *CIKM '07*, pages 909–914, 2007.

[8] J. Leskovec, D. Huttenlocher, and J. Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 641–650, New York, NY, USA, 2010. ACM.

[9] R. Li, S. Wang, H. Deng, R. Wang, and K. C.-C. Chang. Towards social user profiling: unified and discriminative influence model for inferring home locations. In *KDD*, pages 1023–1031, 2012.

[10] Q. Lu and L. Getoor. Link-based classification using labeled and unlabeled data. In *ICML Workshop on "The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining"*, 2003.

[11] S. A. Macskassy and F. Provost. A simple relational classifier. In *Proceedings of the Second Workshop on Multi-Relational Data Mining (MRDM-2003) at KDD-2003*, pages 64–76, 2003.

[12] J. McAuley and J. Leskovec. Learning to discover social circles in ego networks. In P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 548–556. 2012.

[13] A. Mislove, B. Viswanath, K. P. Gummadi, and P. Druschel. You are who you know: inferring user profiles in online social networks. In *WSDM '10*, pages 251–260, 2010.

[14] J. Neville, M. Adler, and D. Jensen. Clustering relational data using attribute and link information. In *Proceedings of the Text Mining and Link Analysis Workshop*, 2003.

[15] M. E. J. Newman. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133, Jun 2004.

[16] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS 01*, pages 849–856. MIT Press, 2001.

[17] S. Nowson and J. Oberlander. The identity of bloggers: Openness and gender in personal weblogs.,. In *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pages 163–167. AAAI, 2006.

[18] M. Pennacchiotti and A.-M. Popescu. Democrats, republicans and starbucks afficionados: user classification in twitter. In *KDD '11*, pages 430–438, 2011.

[19] D. Rao, D. Yarowsky, A. Shreevats, and M. Gupta. Classifying latent user attributes in twitter. In *Proceedings of the 2nd international workshop on Search and mining user-generated contents*, SMUC '10, pages 37–44, New York, NY, USA, 2010. ACM.

[20] J. Tang, S. Wu, and J. Sun. Confluence: conformity influence in large social networks. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '13, 2013.

[21] W. Tang, H. Zhuang, and J. Tang. Learning to infer social ties in large networks. In *ECML/PKDD (3)*, pages 381–397, 2011.

[22] C. Wang, J. Han, Y. Jia, J. Tang, D. Zhang, Y. Yu, and J. Guo. Mining advisor-advisee relationships from research publication networks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 203–212, New York, NY, USA, 2010. ACM.

[23] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Sch?lkopf. Learning with local and global consistency. In *NIPS'04*, pages 321–328. MIT Press, 2004.

[24] Y. Zhou, H. Cheng, and J. X. Yu. Graph clustering based on structural/attribute similarities. *Proc. VLDB Endow.*, 2(1):718–729, Aug. 2009.

[25] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML '03*, pages 912–919, 2003.