Learning to Discover Social Circles in Ego Networks*

Julian McAuley
Stanford
jmcauley@cs.stanford.edu

Jure Leskovec
Stanford
jure@cs.stanford.edu

Abstract

Our personal social networks are big and cluttered, and currently there is no good way to organize them. Social networking sites allow users to manually categorize their friends into *social circles* (e.g. 'circles' on Google+, and 'lists' on Facebook and Twitter), however they are laborious to construct and must be updated whenever a user's network grows. We define a novel machine learning task of identifying users' social circles. We pose the problem as a node clustering problem on a user's ego-network, a network of connections between her friends. We develop a model for detecting circles that combines network structure as well as user profile information. For each circle we learn its members and the circle-specific user profile similarity metric. Modeling node membership to multiple circles allows us to detect overlapping as well as hierarchically nested circles.

1 Introduction

Online social networks allow us to follow streams of posts generated by hundreds of our friends and acquaintances. Our friends generate overwhelming volumes of information and to cope with the 'information overload' we need to organize our personal social networks. One of the main mechanisms for users of social networking sites to organize their networks and the content generated by them is to categorize their friends into what we refer to as *social circles*. Practically all major social networks provide such functionality, for example, 'circles' on Google+, and 'lists' on Facebook and Twitter. Once a user creates her circles, they can be used for content filtering, for privacy, and for sharing groups of users that others may wish to follow.

In this paper we study the problem of automatically discovering users' social circles. In particular, given a single user with her personal social network, our goal is to identify her circles, each of which is a subset of her friends. Circles are user-specific as each user organizes her personal network of friends independently of all other users to whom she is not connected. This means that we can formulate the problem of circle detection as a clustering problem on her ego-network, the network of friendships between her friends. In Figure 1 we are given a single user u and we form a network between her friends v_i. We refer to the user u as the *ego* and to the nodes v_i as *alters*. The task then is to identify the circles to which each alter v_i belongs, as in Figure 1. In other words, the goal is to find nested as well as overlapping communities/clusters in u's ego-network.

We model circle affiliations as latent variables, and similarity between alters as a function of common profile information. We propose an *unsupervised* method to learn which dimensions of profile similarity lead to densely linked circles. Our model has two innovations: First, in contrast to mixed-membership models [2] we predict *hard* assignment of a node to *multiple* circles, which proves critical for good performance. Second, by proposing a parameterized definition of profile similarity, we learn the dimensions of similarity along which links emerge. We achieve this by allowing each circle to have a different definition of profile similarity, so that one circle might form around friends from the same school, and another around friends from the same location.

[0] An extended version of this paper appears in the main conference.

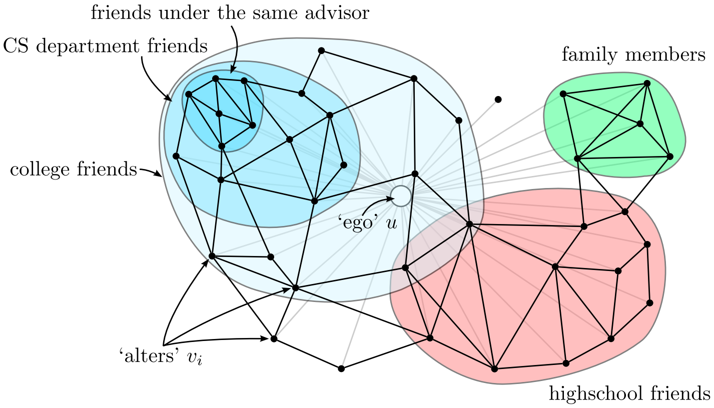


Figure 1: An ego-network with labeled circles. The goal is to discover these circles given the network between the ego's friends. We aim to discover circle memberships and to find common properties around which circles form.

2 A Generative Model for Friendships in Social Circles

We desire a model of circle formation with the following properties: (1) Nodes within circles should have common properties, or 'aspects'. (2) Different circles should be formed by different aspects, e.g. one circle might be formed by family members, and another by students who attended the same university. (3) Circles should be allowed to overlap, and 'stronger' circles should be allowed to form within 'weaker' ones, e.g. a circle of friends from the same degree program may form within a circle from the same university, as in Figure 1. (4) We would like to leverage both profile information and network structure in order to identify the circles. Ideally we would like to be able to pinpoint *which* aspects of a profile caused a circle to form, so that the model is interpretable by the user.

The input to our model is an ego-network $G = (V, E)$, along with 'profiles' for each user $v \in V$. The 'center' node u of the ego-network (the 'ego') is not included in G, but rather G consists only of u's friends (the 'alters'). We define the ego-network in this way precisely because creators of circles do not themselves appear in their own circles. For each ego-network, our goal is to predict a set of circles $\mathcal{C} = \{C_1 \dots C_K\}$, $C_k \subseteq V$, and associated parameter vectors θ_k that encode how each circle emerged. We encode 'user profiles' into pairwise features $\phi(x, y)$ that in some way capture what properties the users x and y have in common.

We describe a model of social circles that treats circle memberships as latent variables. Nodes within a common circle are given an opportunity to form an edge, which naturally leads to hierarchical and overlapping circles.

Our model of social circles is defined as follows. Given an ego-network G and a set of K circles $\mathcal{C} = \{C_1 \dots C_K\}$, we model the probability that a pair of nodes $(x, y) \in V \times V$ form an edge as

$$p((x,y) \in E) \propto \exp\Bigg\{ \underbrace{\sum_{C_k \supseteq \{x,y\}} \langle \phi(x,y), \theta_k \rangle}_{\text{circles containing both nodes}} - \underbrace{\sum_{C_k \not\supseteq \{x,y\}} \alpha_k \langle \phi(x,y), \theta_k \rangle}_{\text{all other circles}} \Bigg\}. \tag{1}$$

For each circle C_k, θ_k is the profile similarity parameter that we will learn. The idea is that $\langle \phi(x, y), \theta_k \rangle$ is high if both nodes belong to C_k, and low if either of them do not (α_k trades-off these two effects). Since the feature vector $\phi(x, y)$ encodes the similarity between the profiles of two users x and y, the parameter vector θ_k encodes what dimensions of profile similarity caused the circle to form, so that nodes within a circle C_k should 'look similar' according to θ_k.

Considering that edges $e = (x, y)$ are generated independently, we can write the probability of G as

$$P_\Theta(G; \mathcal{C}) = \prod_{e \in E} p(e \in E) \times \prod_{e \notin E} p(e \notin E), \tag{2}$$

where $\Theta = \{(\theta_k, \alpha_k)\}^{k=1\dots K}$ is our set of model parameters. Next, we describe how to optimize node circle memberships $\mathcal{C}$ as well as the parameters of the user profile similarity functions $\Theta = \{(\theta_k, \alpha_k)\}$ $(k = 1 \dots K)$ given a graph G and user profiles.

3 Unsupervised Learning of Model Parameters

Treating circles $\mathcal{C}$ as latent variables, we aim to find $\hat{\Theta} = \{\hat{\theta}, \hat{\alpha}\}$ so as to maximize the regularized log-likelihood. We solve this problem using coordinate ascent on Θ and $\mathcal{C}$. For fixed $\mathcal{C} \setminus C_i$ we note that solving $\mathrm{argmax}_{C_i}\, l_\Theta(G; \mathcal{C} \setminus C_i)$ can be expressed as pseudo-boolean optimization in a pairwise graphical model [6]. 'Pseudo-boolean optimization' refers to problems defined over boolean variables (in this case, whether or not a node is assigned to a particular community), where the variables being optimized are interdependent (in this case, relationships are defined over edges in a graph). For further details of our optimization procedure, see the extended version of our paper.

We regularize our model using the ℓ_1 norm, which leads to sparse (and readily interpretable) parameters. Since ego-networks are naturally relatively small, our algorithm can readily handle problems at the scale required. In the case of Facebook, the average ego-network has around 190 nodes [17], while the largest network we encountered has 4,964 nodes. Note that since the method is *unsupervised*, inference is performed independently for each ego-network. This means that our method could be run on the full Facebook graph (for example), as circles are independently detected for each user, and the ego-networks typically contain only hundreds of nodes.

To choose the optimal number of circles, we choose K so as to minimize an approximation to the Bayesian Information Criterion (BIC) [2, 9, 18].

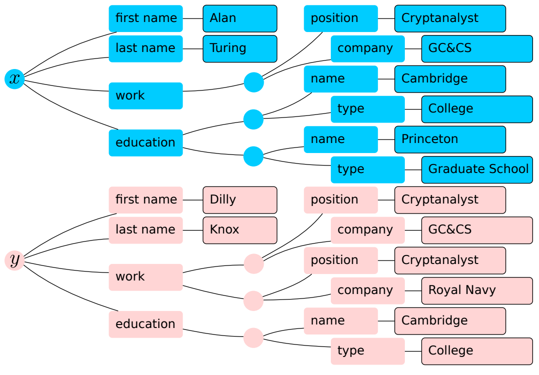
4 Dataset Description

From Facebook we obtained profile and network data from 10 ego-networks, consisting of 193 circles and 4,039 users. To obtain circle information we developed our own Facebook application and conducted a survey of ten users, who were asked to manually identify all the circles to which their friends belonged. It took each user between 2 and 3 hours to label their entire network. On average, users identified 19 circles in their ego-networks, with an average circle size of 22 friends. Examples of circles we obtained include students of common universities and classes, sports teams, relatives, etc. For the other two datasets we obtained publicly accessible data. From Google+ we obtained data from 133 ego-networks, consisting of 479 circles and 106,674 users. The 133 ego-networks represent all 133 Google+ users who had shared at least two circles, and whose network information was publicly accessible at the time of our crawl. The Google+ circles are quite different to those from Facebook, in the sense that their creators have chosen to release them publicly, and because Google+ is a *directed* network (note that our model can very naturally be applied to both to directed and undirected networks). Finally, from Twitter we obtained data from 1,000 ego-networks, consisting of 4,869 circles (or 'lists' [11, 13, 19, 21]) and 81,362 users. The ego-networks we obtained range in size from 10 to 4,964 nodes.

5 Constructing Features from User Profiles

From Google+ we collect data from six categories (gender, last name, job titles, institutions, universities, and places lived). From Facebook we collect data from 26 categories, including users' hometowns, birthdays, colleagues, political and religious affiliations, etc. As a proxy for profile data, from Twitter we collect data from two categories, namely the set of hashtags and mentions used by each user during two-weeks' worth of tweets. 'Categories' correspond to parents of leaf nodes in a profile tree, as shown in Figure 2.

We first propose a difference vector to encode the relationship between two profiles. A non-technical description is given in Figure 2. Suppose that users $v \in V$ each have an associated profile tree T_v, and that $l \in T_v$ is a leaf in that tree. We define the difference vector $\sigma_{x,y}$ between two users x and y as a binary indicator encoding the profile aspects where users x and y differ (Figure 2, top right). Although such a difference vector has the advantage that it encodes profile information at a fine granularity, it has the disadvantage that it is high-dimensional (up to 4,122 dimensions in the data we considered). One way to address this is to form difference vectors based on the *parents* of leaf nodes: this way, we encode what profile *categories* two users have in common, but disregard specific values (Figure 2, bottom right).



$$\mathbf{1} - \sigma_{x,y} = \left[\begin{array}{c}0\\0\\0\\0\\1\\1\\0\\1\\1\\0\\0\end{array}\right] \begin{array}{l} \mathit{first\ name : Dilly}\\ \mathit{last\ name : Knox}\\ \mathit{first\ name : Alan}\\ \mathit{last\ name : Turing}\\ \mathit{work : position : Cryptanalyst}\\ \mathit{work : location : GC\&CS}\\ \mathit{work : location : Royal\ Navy}\\ \mathit{education : name : Cambridge}\\ \mathit{education : type : College}\\ \mathit{education : name : Princeton}\\ \mathit{education : type : Graduate\ School}\end{array}$$

$$\mathbf{1} - \sigma'_{x,y} = \left[\begin{array}{c}0\\0\\1\\1\\1\\1\end{array}\right] \begin{array}{l} \mathit{first\ name}\\ \mathit{last\ name}\\ \mathit{work : position}\\ \mathit{work : location}\\ \mathit{education : name}\\ \mathit{education : type}\end{array}$$

Figure 2: Feature construction. Profiles are tree-structured, and we construct features by comparing paths in those trees. Examples of trees for two users x (blue) and y (pink) are shown at left. Two schemes for constructing feature vectors from these profiles are shown at right: (1) (top right) we construct binary indicators measuring the difference between leaves in the two trees, e.g. 'work→position→Cryptanalyst' appears in both trees. (2) (bottom right) we sum over the leaf nodes in the first scheme, maintaining the fact that the two users worked at the same institution, but discarding the *identity* of that institution.

Based on the difference vectors $\sigma_{x,y}$ (and $\sigma'_{x,y}$) we now describe how to construct edge features $\phi(x, y)$. The first property we wish to model is that *members of circles should have common relationships* with each other:

$$\phi^1(x, y) = (1; -\sigma_{x,y}). \tag{3}$$

The second property we wish to model is that *members of circles should have common relationships to the ego of the ego-network.* In this case, we consider the profile tree T_u *from the ego user* u. We then define our features in terms of that user:

$$\phi^2(x, y) = (1; -\big|\sigma_{x,u} - \sigma_{y,u}\big|) \tag{4}$$

($|\sigma_{x,u} - \sigma_{y,u}|$ is taken elementwise). These two parameterizations allow us to assess which mechanism better captures users' subjective definition of a circle. In both cases, we include a constant feature ('1'), which controls the probability that edges form within circles, or equivalently it measures the extent to which circles are made up of friends. Importantly, this allows us to predict memberships even for users who have no profile information, simply due to their patterns of connectivity.

Similarly, for the 'compressed' difference vector $\sigma'_{x,y}$, we define

$$\psi^1(x, y) = (1; -\sigma'_{x,y}), \quad \psi^2(x, y) = (1; -\big|\sigma'_{x,u} - \sigma'_{y,u}\big|). \tag{5}$$

6 Extensions

In this section, we describe techniques to exploit partially observed circle information to help users update and maintain their circles. In other words, we would like to apply our model to users' personal networks as they change and evolve [4]. Since our model is probabilistic, it is straightforward to adapt it to make use of partially observed data, by conditioning on the assignments of some of the latent variables in our model. In this way, we adapt our model for semi-supervised settings in which a user labels some or all of the members of their circles.

6.1 Circle Maintenance

First we deal with the problem of a user adding new friends to an established ego-network, whose circles have already been defined. Thus, given a complete set of circles, our goal is to predict community memberships for a new node, based on that node's features, and their patterns of connectivity to existing nodes in the ego-network.

Since circles in this setting are fully-observed, we simply fit the model parameters that best explain the ground-truth circles $\bar{\mathcal{C}}$ provided by the user. Optimization is significantly faster in this case as there are no longer latent community memberships to infer.

Next, we must predict to which of the K ground-truth circles a new user u belongs. That is, we must predict $c^u \in \{0, 1\}^K$, where each c^u_k is a binary variable indicating whether the user u should belong to the circle C_k. In practice, for the sake of evaluation, we shall suppress a single user from G and $\bar{\mathcal{C}}$, and try to recover their memberships.

This can be done by choosing the assignment c^u that maximizes the log-likelihood of $\mathcal{C}$ once u is added to the graph. This can be computed efficiently for different values of c^u by noting that the log-likelihood only changes for terms including u, meaning that we need to compute $p((x, y) \in E)$ only if $x = u$ or $y = u$. In other words, we only need to consider how the new user relates to existing users, rather than considering how existing users relate to each other; thus computing the log-likelihood requires linear (rather than quadratic) time. To find the optimal c^u we can simply enumerate all 2^K possibilities, which is feasible so long as the user has no more than $K \simeq 20$ circles. For users with more circles we must resort to an iterative update scheme as we did in Section 3.

6.2 Semi-Supervised Circle Prediction

Next, we consider the problem of using weak supervision in the form of 'seed nodes' to assist in circle prediction [3]. In this setting, the user manually labels a few users from each of the circles they want to create, say $\{s_1 \dots s_K\}$. Our goal is then to predict K circles $\mathcal{C} = \{C_1 \dots C_K\}$ subject to the constraint that $s_k \subseteq C_k$ for all $k \in \{1 \dots K\}$.

Again, since our model is probabilistic, this can be done by conditioning on the assignments of some of the latent variables. That is, we simply optimize the log-likelihood subject to the constraint that $s_k \subseteq C_k$ for all $k \in \{1 \dots K\}$. In the parlance of graphical models, this means that rather than treating the seed nodes as latent variables to be predicted, we treat them as evidence on which we condition. We could also include negative evidence (i.e., the user could provide labels for users who do *not* belong to each circle), or we could have users provide additional labels interactively, though the setting described is the most similar to what is used in practice.

7 Experiments

Although our method is unsupervised, we can evaluate it on ground-truth data by examining the maximum-likelihood assignments of the latent circles $\mathcal{C} = \{C_1 \dots C_K\}$ after convergence. Our goal is that for a properly regularized model, the latent circles will align closely with the human labeled ground-truth circles $\bar{\mathcal{C}} = \{\bar{C}_1 \dots \bar{C}_{\bar{K}}\}$.

To measure the alignment between a predicted circle C and a ground-truth circle $\bar{C}$, we compute the Balanced Error Rate (BER) between the two circles. This measure assigns equal importance to false positives and false negatives, so that trivial or random predictions incur an error of 0.5 on average.

7.1 Aligning predicted and ground-truth circles

Since we do not know the correspondence between circles in $\mathcal{C}$ and $\bar{\mathcal{C}}$, we compute the optimal match via linear assignment by maximizing:

$$\max_{f:\mathcal{C}\to\bar{\mathcal{C}}} \frac{1}{|f|} \sum_{C \in \text{dom}(f)} (1 - BER(C, f(C))), \tag{6}$$

where f is a (partial) correspondence between $\mathcal{C}$ and $\bar{\mathcal{C}}$. That is, if the number of predicted circles $|\mathcal{C}|$ is less than the number of ground-truth circles $|\bar{\mathcal{C}}|$, then every circle $C \in \mathcal{C}$ must have a match $\bar{C} \in \bar{\mathcal{C}}$, but if $|\mathcal{C}| > |\bar{\mathcal{C}}|$, we do not incur a penalty for additional predictions that *could* have been circles but were not included in the ground-truth. We use established techniques to estimate the number of circles, so that none of the baselines suffers a disadvantage by mispredicting $\hat{K} = |\mathcal{C}|$.

Baselines. We considered a wide number of baseline methods, including those that consider only network structure, those that consider only profile information, and those that consider both. First we experimented with *Mixed Membership Stochastic Block Models* [2], which consider only network information, and variants that also consider text attributes [7, 8, 12]. For each node, mixed-membership models predict a stochastic vector encoding partial circle memberships, which we

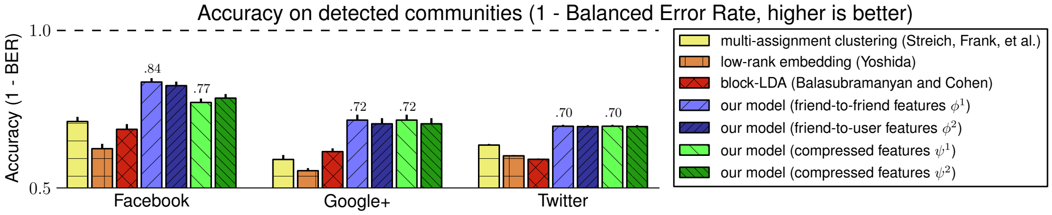


Figure 3: Performance on Facebook, Google+, and Twitter. Higher is better.

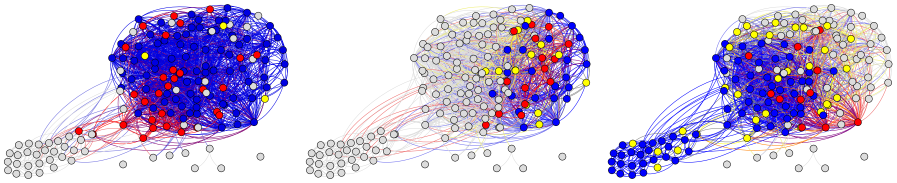


Figure 4: Three detected circles on a small ego-network from Facebook, compared to three ground-truth circles (BER $\simeq 0.81$). Blue nodes: true positives. Grey: true negatives. Red: false positives. Yellow: false negatives. Our method correctly identifies the largest circle (left), a sub-circle contained within it (center), and a third circle that significantly overlaps with it (right).

threshold to generate 'hard' assignments. We also considered *Block-LDA* [5], where we generate 'documents' by treating aspects of user profiles as words in a bag-of-words model.

Secondly, we experimented with classical clustering algorithms, such as *K-means* and *Hierarchical Clustering* [10], that form clusters based only on node profiles, but ignore the network. Conversely we considered *Link Clustering* [1] and *Clique Percolation* [15], which use network information, but ignore profiles. We also considered the *Low-Rank Embedding* approach of [20], where node attributes *and* edge information are projected into a feature space where classical clustering techniques can be applied. Finally we considered *Multi-Assignment Clustering* [16], which is promising in that it predicts hard assignments to multiple clusters, though it does so without using the network.

Of the eight baselines highlighted above we report the three whose overall performance was the best, namely Block-LDA [5] (which slightly outperformed mixed membership stochastic block models [2]), Low-Rank Embedding [20], and Multi-Assignment Clustering [16].

7.2 Performance on Facebook, Google+, and Twitter Data

Figure 3 shows results on our Facebook, Google+, and Twitter data. The largest circles from Google+ were excluded as they exhausted the memory requirements of many of the baseline algorithms. Circles were aligned as described in (eq. 6), with the number of circles $\hat{K}$ determined as described in Section 3. For non-probabilistic baselines, we chose $\hat{K}$ so as to maximize the *modularity* [14]. In terms of absolute performance our best model ϕ^1 achieves BER scores of 0.84 on Facebook, 0.72 on Google+ and 0.70 on Twitter (F_1 scores are 0.59, 0.38, and 0.34, respectively).

Comparing our method to baselines we notice that we outperform all baselines on all datasets by a statistically significant margin. Compared to the nearest competitors, our best performing features ϕ^1 improve on the BER by 43% on Facebook, 26% on Google+, and 16% on Twitter (improvements in terms of the F_1 score are similar). Regarding the performance of the baseline methods, we note that good performance seems to depend critically on predicting *hard* memberships to *multiple* circles, using a combination of *node and edge* information; none of the baselines exhibit precisely this combination, a shortcoming our model addresses.

Examining the output of our model in greater detail, Figure 4 shows results of our unsupervised method on example ego-networks from Facebook. Different colors indicate true-, false- positives

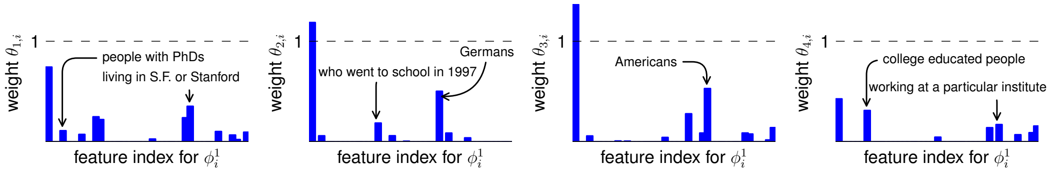


Figure 5: Parameter vectors of four communities for a particular Facebook user.

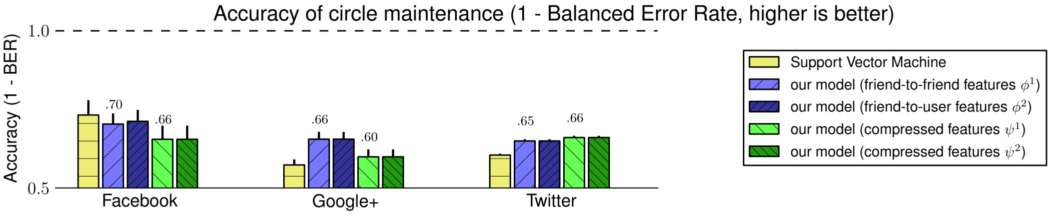


Figure 6: Accuracy of assigning a new node to already-existing circles. Although a fully-supervised Support Vector Machine gives accurate results on Facebook (where node features are highly informative), our model yields far better results on Google+ and Twitter data. Results in terms of the $F-1$ score are qualatively similar.

and negatives. Our method is correctly able to identify overlapping circles as well as sub-circles (circles within circles). Figure 5 shows parameter vectors learned for four circles for a particular Facebook user. Positive weights indicate properties that users in a particular circle have in common. Notice how the model naturally learns the social dimensions that lead to a social circle. Moreover, the first parameter that corresponds to a constant feature '1' has the highest weight; this reveals that membership to the same community provides the strongest signal that edges will form, while profile data provides a weaker (but still relevant) signal.

7.3 Circle Maintenance

Next we examine the problem of adding new users to already-defined ego-networks, in which complete circles have already been provided. For evaluation, we suppress a single user u from a user's ego-network, and learn the model parameters $\hat{\Theta}$ that best fit $G \setminus \{u\}$ and $\mathcal{C} \setminus \{u\}$. Our goal is then to recover the set of communities to which the node u belongs, as described in Section 6.1. Again we report the Balanced Error Rate between the ground-truth and the predicted set of community memberships for u. We use all of each users' circles for training, up to a maximum of fifteen circles. This experiment is repeated for 10 random choices of the user u for each ego-network in our dataset.

Performance on this task is shown in Figure 6. On Facebook, Google+, and Twitter our best performing features ϕ^1 achieve Balanced Error Rates of 0.30, 0.34, and 0.34 (respectively), and F_1 scores of 0.38, 0.59, and 0.54. A fully-supervised SVM baseline achieves better accuracy when rich node features are available (which is the case for Facebook), though it fails to make use of edge information, and does not account for interdependencies between circles. This proves critical in the case of Google+ and Twitter, where node information alone proves uninformative.

7.4 Semi-Supervised Circle Prediction

Our final task is to identify circles using a form of weak supervision provided by the user, in the form of *seed nodes* as described in Section 6.2. In this setting, the user provides S seed nodes for each of K circles that they wish to identify. For evaluation, we select the K circles to be identified and the S seed nodes uniformly at random.

Figure 7 shows the performance of our algorithm for different numbers of seed nodes $S \in \{0 \dots 10\}$ and different numbers of circles $K \in \{1 \dots 5\}$ on Facebook. We find that for all values of K,

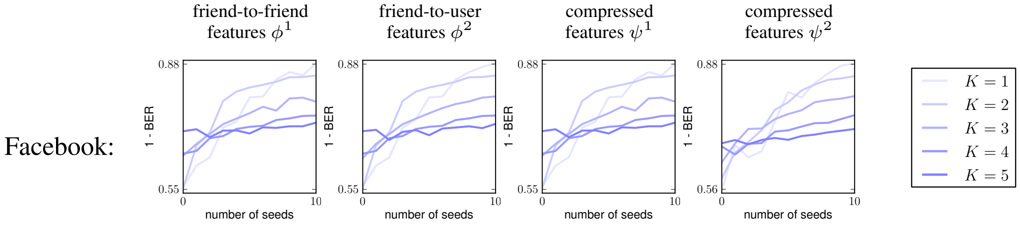


Figure 7: Number of seeds versus accuracy for different numbers of circles K. For each of the K circles being identified, the user provides the same number of seeds. Results on Google+ and Twitter are qualitatively similar and are omitted for brevity.

adding seed nodes increases the accuracy significantly, though the effect is most pronounced when the number of circles that the user wishes to identify is small.

References

[1] Y.-Y. Ahn, J. Bagrow, and S. Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 2010.

[2] E. Airoldi, D. Blei, S. Fienberg, and E. Xing. Mixed membership stochastic blockmodels. *Journal of Machine Learning Research*, 2008.

[3] R. Andersen and K. Lang. Communities from seed sets. In *WWW*, 2006.

[4] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

[5] R. Balasubramanyan and W. Cohen. Block-LDA: Jointly modeling entity-annotated text and entity-entity links. In *SIAM International Conference on Data Mining*, 2011.

[6] E. Boros and P. Hammer. Pseudo-boolean optimization. *Discrete Applied Mathematics*, 2002.

[7] J. Chang and D. Blei. Relational topic models for document networks. In *International Conference on Artificial Intelligence and Statistics*, 2009.

[8] J. Chang, J. Boyd-Graber, and D. Blei. Connections between the lines: augmenting social networks with text. In *Knowledge Discovery and Data Mining*, 2009.

[9] M. Handcock, A. Raftery, and J. Tantrum. Model-based clustering for social networks. *Journal of the Royal Statistical Society Series A*, 2007.

[10] S. Johnson. Hierarchical clustering schemes. *Psychometrika*, 1967.

[11] D. Kim, Y. Jo, L.-C. Moon, and A. Oh. Analysis of twitter lists as a potential source for discovering latent characteristics of users. In *CHI*, 2010.

[12] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link LDA: joint models of topic and author community. In *International Conference on Machine Learning*, 2009.

[13] P. Nasirifard and C. Hayes. Tadvise: A twitter assistant based on twitter lists. In *SocInfo*, 2011.

[14] M. E. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 2006.

[15] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 2005.

[16] A. Streich, M. Frank, D. Basin, and J. Buhmann. Multi-assignment clustering for boolean data. In *International Conference on Machine Learning*, 2009.

[17] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. The anatomy of the Facebook social graph. preprint, 2011.

[18] C. Volinsky and A. Raftery. Bayesian information criterion for censored survival models. *Biometrics*, 2000.

[19] S. Wu, J. Hofman, W. Mason, and D. Watts. Who says what to whom on twitter. In *WWW*, 2011.

[20] T. Yoshida. Toward finding hidden communities based on user profiles. In *ICDM Workshops*, 2010.

[21] J. Zhao. Examining the evolution of networks based on lists in twitter. In *International Conference on Internet Multimedia System Architectures and Applications*, 2011.