Dynamic Circle Recommendation: A Probabilistic Model

Fan-Kai Chou[1], Meng-Fen Chiang[1], Yi-Cheng Chen[2], and Wen-Chih Peng[1]

[1] Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan
[2] Department of Computer Science and Information Engineering, Tamkang University, Taiwan
{plapla.cs00g,mfchiang.cs95g}@nctu.edu.tw, ycchen@mail.tku.edu.tw, wcpeng@cs.nctu.edu.tw

Abstract. This paper presents a novel framework for dynamic circle recommendation for a query user at a given time point from historical communication logs. We identify the fundamental factors that govern interactions and aim to automatically form dynamic circle for scenarios, such as, *who should I dial to in the early morning? whose mail would I reply first at midnight?* We develop a time-sensitive probabilistic model (TCircleRank) that not only captures temporal tendencies between the query user and candidate friends but also blends frequency and recency into group formation. We also utilize the model to support two types of dynamic circle recommendation: **Seedset Generation**: single-interaction suggestion and **Circle Suggestion**: multiple interactions suggestion. We further present approaches to infer relevant time interval in determining circles for a query user at a given time. Experimental results on Enron dataset, Call Detail Records and Reality Mining Data prove the effectiveness of dynamic circle recommendation using TCircleRank.

1 Introduction

As the emergence of on-line social media, users can easily share information to their friends via Mobile Social Media Apps such as Gmail, WhatsApp, Facebook using their mobile devices. Social media gather and syndicate these information to target users. Users can browse through the information shared by their friends. Most existing social media generally render information based on recency, that is, latest information always appear on top of personal feed walls. Some may provide manual tools for users to explicitly adjust friend circles so that users can control how information are rendered on their walls or which friend circles to share information with. Such great efforts motivate us to wonder: Is it possible to design a dynamic circle recommendation system which can automatically suggest a ranked list of friend candidates driven by both historical interaction statistics and contextual information such as time point?

Most studies on formation of groups mainly focus on static group formation, where a group is a fixed set of friedns manually pre-defined by a user. We argue

V.S. Tseng et al. (Eds.): PAKDD 2014, Part II, LNAI 8444, pp. 25–37, 2014.

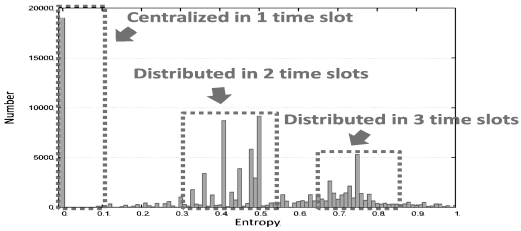


Fig. 1. Distribution of Time Centrality

that the notion of group should dynamically adapt to context information such as location, time, etc. That is, some users may have the tendency to share information to different groups of friends at certain time points while some users may share information to the same group of friends at all time. For example, a user may have the tendency to share information to his/her family during daytime and share information to his/her close colleagues in the evening. To discover time-dependency for a target user, we need to identify his/her tendency at different time point. Following this, we need to provide a ranked list of friends that a user has the highest probability to interact with at each time point.

The general problem of recommendation system has been widely studied [6]. Several prior studies attempt to consider temporal factor in designing recommendation systems [9][3][2][11]. For example, [9] leveraged user's long-term and short-term preferences for temporal recommendation. Nonetheless, non of them addresses the fact that user interactions are not always correlate with time as users present diverse variation of temporal dependency. For example, some users have higher temporal dependency in sharing information. Moreover, a user may only be sensitive to certain time points during a day. In this paper, we argue that temporal tendency should be analyzed individually for each pair of query user and friend candidate at each time point. As an evidence, Figure 1 illustrates a distribution of time centrality for all pairs of users. If a pair of users' interactions only fall into a few time slots during a day, they have lower entropy and thus indicating higher time centrality and vice versa. We observe that over 60% pairs of users' have higher time centrality in interactions (entropy ≤ 0.5), meaning the rest 40% user interactions are driven or dominated by other factors.

In this paper, we propose a framework to discover personalized dynamic circle for a given time point. Given a query user, a time point, and historical communication logs, our recommendation system returns a ranked list of friends (referred to as Circle) for the query user at given time point. To achieve this, we propose a temporal probabilistic model (*TCircleRank*) to capture user behaviors in terms of three factors: frequency, recency and time-dependency. After this, we utilize TCircleRank to derive two types of dynamic circle recommendation: **Seedset Generation**: single interaction suggestion and **Circle Suggestion**: multiple interactions suggestion. TCircleRank considers the dynamic importance of each candidate user for a query user to incorporate the factor, *different users show different temporal dependency with related to a target user at different time.*

Seedset Generation aims to generate a candidate user by TCircleRank for two purposes: shifting the burden for query users (especially mobile users) to provide a list of users who intent to interact with at the very beginning, and the query user merely interact with a single user at given time.

Recommending dynamic circle is useful in many applications. For example, dynamic circle can be utilized to enhance the ranking results for content-based on-line social media (e.g., Gmail, WhatsApp, Facebook), where the information for each user can be adjusted based on the dynamic circle. Moreover, it can be used in location sharing services (e.g., Foursquare), where the ranking of locations can be adjusted based on a user's dynamic circle at particular time point. To summarize, our contributions are as follows.

- We propose a framework to discover personalized dynamic circle for a query user at given time point.
- We propose a temporal probabilistic model (TCircleRank) to capture user's interaction tendency at different time point.
- We consider three fundamental factors in user interactions and propose approaches to support: single interaction suggestion and multiple interactions suggestion.
- We proposes two methods to find the most appropriate time interval for our probabilistic model.
- We conduct experiments on real datasets to demonstrate the effectiveness of our framework and report empirical insights.

This paper is organized as follows. Section 2 presents the related work for this paper. Section 3 introduces TCircleRank and then discusses the two types of dynamic circle recommendation. Section 4 presents two methods to infer time interval for TCircleRank. Section 5 shows the experimental results using the three real datasets. Section 6 concludes this paper.

2 Related Work

2.1 Relationship Link Prediction

Friends suggestion can be modeled as relationship link prediction, if we predict the occurrence of an interaction at a given time. Liben-Nowell and Kleinberg [4] formalized the link prediction problem and employed random walk methods to address this problem. Yang *et al.* [10] proposed FIP model bridges between Collaborative Filtering (CF) and link prediction to provide a unified treatment for interest targeting and friendship prediction. Sun *et al.* [7] built a relationship building time prediction model, which uses learning algorithms to fit different distributions and then gets a probability for building relationships between two nodes. However, the edges are only constructed once, so we cannot use it for communication networks which change over time.

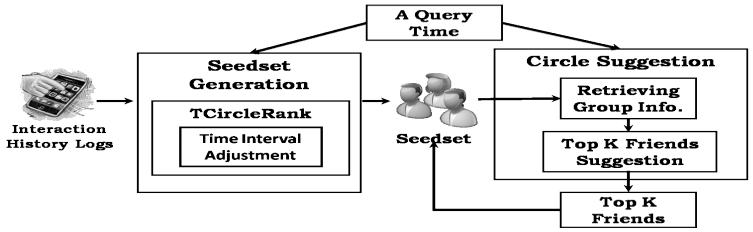


Fig. 2. Framework Overview

2.2 Friends Suggestion System

Our main idea is based on Roth *et al.* [5], who proposed a friends recommendation system for *Gmail* using group information and three criteria. *Gmail* is a well-known mail system constructed by *Google*, which may have many history records to retrieve for friends suggestion. However, the algorithm in [5] could not work effectively for sparse data, insufficient interaction history resulting in some recommendation lists to be empty. Moreover, *Time-Dependency* of user interactions is not addressed in their work. Bartel and Dewan [1] enhanced [5] with a hierarchical structure, which re-orders the recommendation list by ranking past communication group and hierarchically predicts next group. Wu *et al.* [8] proposed a interactive learning framework to formulate the problem of recommending patent partners into a factor graph model. Similarly, no attention has been paid to address the problem of *Time-Dependency* of user interactions.

3 Dynamic Circle Recommendation

We propose a framework for dynamic circle recommendation without requiring query users to provide any information as a prior. The system framework overview is illustrated in Fig. 2. Our system consists of two phases: Seedset Generation and Circle Suggestion. Seedset Generation automatically derives a set of core users (*referred to as seedset*) with the highest probability to be contacted with the query user. Seedset Generation is achieved by mining frequent and time-dependent communication patterns from historical interaction logs. Circle Suggestion phase aims to provide a group of friends whenever the query user intends to interact with multiple users at the same time (*referred to as circle*) based on the derived seedset. Once the query user chooses partial members from the list, our system updates the circle suggestion list by adding selected users to current seedset and then launching Circle Suggestion again to update the ranked list of friends. This process continues until no more friends can be suggested or the query user drops this session. Notice that our recommendation system provides a generic framework where the Circle Suggestion component can be replaced by other state-of-the-art algorithms to serve different requirements.

3.1 TCircleRank

When a query user attempts to share information (e.g., photos), the query user forms a list of friends in his/her mind. Without any assistance, query users has to manually and sequentially select the list of friends by scanning through their friends pool. This brings lots of unnecessary efforts. To solve this problem, we first propose Seedset Generation that uses TCircleRank to predict a set of users as seeds for Circle Suggestion.

We claim that *if a query user interacts with a user in a particular time interval, the query user has a higher probability to interact with the user in the time interval as well.* Fig. 1 verifies this by showing that, over 60% of interactions are strongly temporal-correlated with entropy is no greater than 0.5.

To address this, we propose a framework, TCircleRank, to predict a ranked list of friends who are most likely to be interacted with the query user a given time point. There are three factors considered in TCircleRank:

1. Frequency: Receivers who have more interactions with the query user are more important than those who interact less with the query user.
2. Recency: More recent interactions should have more importance whereas older interactions decay over time.
3. Time-Dependency: If receivers always interact with the query user in a similar time interval, they should have more importance in that time interval.

Frequency is a straightforward yet effective measurement. Inspired from Interaction Rank [5], we unify *Frequency* with *Recency* into a single measurement as shown in Equation (1). [5] introduced a decaying parameter λ, to control the importance of every interaction according to its time. Namely, every interaction decays exponentiation over time with a half life λ. To fit TCircleRank, we form the two factors into a probability, which can be expressed as:

$$P(R_n) = \frac{\sum_{i \in I(R_n)} (\frac{1}{2})^d}{\sum_{i \in I} (\frac{1}{2})^d} \tag{1}$$

where $P(R_n)$ is the probability of the query user interacting with R_n in the past, I is a set of all the query user's interactions, and $I(R_n)$ is a set of all interactions between query user and R_n. d is a decay function which is expressed as $\frac{t_{now} - t_i}{\lambda}$, where t_{now} is the current time, t_i is the time of interaction $i \in I$, and a half-life parameter λ that assigns score 1 to an interaction at current time and decays the importance of an interaction to $\frac{1}{2}$ with the half-life λ.

To incorporate the third factor, *Time-Dependency*, we formulate a conditional probability as:

$$P(R_n|t) = \frac{P(R_n \cap t)}{P(t)}. \tag{2}$$

Equation (2) shows the probability of the query user interacting with R_n in a time interval t, where $P(R_n \cap t)$ and $P(t)$ can be derived like Equation 1 if we change $I(R_n)$ to $I(R_n \cap t)$ and $I(t)$.

To take into the following three factors into consideration, *Frequency*, *Recency* and *Time-Dependency*. Intuitively, we combine $P(R_n)$ and $P(R_n|t)$ by a linear combination with a tunable parameter α, which can be formulated as follows:

$$Score(R_n) = (1 - \alpha)P(R_n) + \alpha \cdot P(R_n|t) \tag{3}$$

where α is the weight of *Time-Dependency* and the range of α is between 0 and 1. In general, Equation (3) does not make sense, because when a candidate receiver R_n has higher $P(R_n)$ and also has higher $P(R_n|t)$, it should be chosen with more chances. When both probabilities are not relative to each other, we should think about other methods to merge them. Calculating the mean between $P(R_n)$ and $P(R_n|t)$ is a good idea to balance Equation (3), because it considers the influence from not only specific time intervals but also all time intervals. We adjust Equation (3) by using geometric mean, and thus the equations can be expressed as follows:

$$Score_{geo}(R_n) = (1 - \alpha)P(R_n) + \alpha \cdot \sqrt[1+\omega]{P(R_n)(P(R_n|t))^{\omega}} \tag{4}$$

where ω represents the weight of a specific time interval. We find that geometric mean makes sense for our assumption: *if one of* $P(R_n)$ *and* $P(R_n|t)$ *is much lower than the other, their mean should be closer to the lower one.*

To refine Equation (4), we need to define the best α. According to our observations, we find that not all receivers have high time-dependency, as some shows similar behaviors regardless of any time points. In other words, receivers have different time-dependencies in different time intervals. Thus, time-dependencies will vary from person to person. To achieve this, we change α to another conditional probability, $P(t|R_n)$, which is indicates the probability of R_n interacting with the query user in time interval t. If $P(t|R_n)$ is higher, R_n has a higher time-dependency with the query user and vice versa. We then utilize Z-score to normalize importance of time-dependency. Because Z-score may be negative, we normalize Z-score by considering the central point from the range [-3, 3] to [0, 1]. Therefore, we can reformulate Equation (4) as follows:

$$\begin{aligned} Score_{final}(R_n) = {} & (1 - NZ(R_n)) \cdot P(R_n) \\ & + NZ(R_n) \cdot \sqrt[1+\omega]{P(R_n)(P(R_n|t))^{\omega}} \end{aligned} \tag{5}$$

where $NZ(R_n)$ is the normalized Z-score and the range is from 0 to 1.

3.2 Seedset Generation

Seedset Generation phase derives a set of core friends who are most likely to be the receivers with related to the query user at given time. In a sense, Seedset Generation can serve as a Circle Suggestion in a special case when query users intend to communicate with a single user instead of a group of users. In that case, Seedset Generation phase returns the potential receivers as a top-k list of users.

Without specific groups information, Seedset Generation adopts TCircleRank mentioned before predicting which friends in the past are most likely to be the receivers, merely based on specified query time. The algorithm of Seedset Generation is summarized in Algorithm 1.

Algorithm 1. Seedset Generation Algorithm

Input: query user's history interactions I and current time interval t
Output: a set of core friends S

```
1  S = φ;
2  foreach i ∈ I do
3      Sum scores of i for TCircleRank;
4      C = GetFriend(i);
5      foreach c ∈ C do
6          if c ∉ S then
7              Put c into S;
8  foreach c ∈ S do
9      Calculate all probabilities P(c), P(t), P(c|t) and P(t|c);
10     S[c] = Score_final(c);
```

3.3 Circle Suggestion

Circle Suggestion can be applied to any seed-based suggestion approach. In this subsection, we propose an enhanced approach, Circle Suggestion, by incorporating the state-of-the-art ranking model [5] with TCircleRank.

TCircleRank can be combined with *Interaction Rank* [5]. *Interaction Rank* only considered three factors, *Frequency*, *Recency* and *Direction*, and we consider one additional factor, *Time-Dependency*. *Interaction Rank* is formally defined as follows:

$$\mathcal{IR}(g) = \theta_{out} \sum_{i \in I_{out}(g)} (\frac{1}{2})^d + \sum_{i \in I_{in}(g)} (\frac{1}{2})^d \tag{6}$$

where $I_{out}(g)$ is the set of outgoing interactions between a query user and a group, $I_{in}(g)$ is the set of incoming interactions and θ_{out} is the weight of outgoing interactions to represent *Direction*. To form a circle of friends, we adopt *Intersection Weighed Score*, which considers the intersection of group and seedset to weight the score of the group. As reported in [5], *Interaction Weighted Score* achieves the best performance among their proposals.

4 Time Interval Adjustment

Considering the following scenario: *A user A has a regular behavior to call user B after user A finishes his works during 5:00pm and 6:00pm. One day, user A has finished his works early at 3:30pm and he calls user B immediately.* Should the interaction at 3:30pm be considered as reference interactions in suggesting

friends? To answer this question, we propose two approaches to identify the time intervals as references in ranking friends. The main idea is to analyze the time distribution of interactions in one day and then determine an optimal time interval to describe the interaction behaviors.

4.1 Entropy Examination

We utilize entropy as a measurement to determine the optimal time interval. A narrow time interval indicates regular behavior and a broad time interval indicates relatively irregular behavior. To measure the regularity of user behaviors, we start with 24 time slots and calculate the entropy for user interactions across each time slots. If the entropy is lower than a threshold, which means the level of regularity is higher enough, we choose $(h-1)/2$ as the optimal time interval, where h is the length of each time slot. Otherwise, we continue to split 24 hours into 16, 12, 8, 6 or 4 time slots until the entropy is lower than a threshold.

4.2 Close Peak Detection

To detect the close peak, we only need to know the trends between each time slot. The goal is to find the cluster that contains the current time slot and then we can choose this cluster as optimal time interval. First, we consider the trend between two adjacent time slots. Larger number of interactions time slot should be less or equal τ times than smaller number of interactions time slot, where τ is a threshold for clustering time slots. Otherwise, the detection would be terminated and the final cluster has been determined. Algorithm 2 describes Close Peak Detection in detail.

Algorithm 2. Close Peak Detection

Input: Time Distribution in 24 hours D, Current Time h and Threshold τ
Output: Time Interval Start T_s and Time Interval End T_e

```
1  T_s = h;
2  T_e = h;
3  foreach Clockwise Time Slots: T_e, x_2 ∈ D do
4      if p(x_2) > p(T_e)&p(x_2) ≤ τ * p(T_e) then
5          T_e = x_2;
6      else if p(x_2) < p(T_e)&p(T_e) ≤ τ * p(x_2) then
7          T_e = x_2;
8  foreach Counterclockwise Time Slots: T_s, x_2 ∈ D do
9      if p(x_2) > p(T_s)&p(x_2) ≤ τ * p(T_s) then
10         T_s = x_2;
11     else if p(x_2) < p(T_s)&p(T_s) ≤ τ * p(x_2) then
12         T_s = x_2;
```

5 Experiment

5.1 Datasets

Social interactions present in calling and mailing behaviors. Therefore, we use calling behavior and mailing behavior datasets to simulate general social behavior dataset. In our experiment, we use three real datasets, Enron Mail[1], call detail records (cdr) from Chunghwa Telecom (CHT)[2] and Reality Mining Dataset (RMD) from MIT[3]. The basic information of each dataset is shown in Table 1, where Enron Mail contains multiple interaction data and the others only contains single interaction data. Therefore, we adopt Enron Mail to evaluate Seedset Generation and Circle Suggestion, and the others two dataset to evalute Seedset Generation.

Table 1. Basic Information on the Enron/CHT/RMD Datasets

Element	Enron	CHT	RMD
No. of user	65,182	76,263	92
No. of interactions	236,505	2,443,667	78,110
No. of group interactions	67,631	-	-
time	1998/01/04 - 2002/12/21	2010/08	2004/01/19 - 2005/07/15

5.2 Time Centrality Analysis

In time centrality experiment, we constrained the number of interaction between the user and the test query user exceeds four times, because we split the time of one day into four time slots of six hours.

We observe the difference of time centrality distribution between Enron and other datasets on Fig. 3 and find that Enron Mail has higher time centrality because its entropy is relatively lower than those of CHT and RMD. This indicates that mailing behavior is relatively regular for the same receiver, i.e., most user tend to send their mail to the same receiver at particular time points. Unlike mailing behavior, calling behavior does not show strong time centrality. The calling behaviors in CHT and RMD are similar and they distribute around entropy 0.5. This explains that when the entropy is 0.5, the users call callees not only at the same time slot but also at the adjacent time slots. In other words, the regular calling behavior may shift to the temporally close time points occasionally.

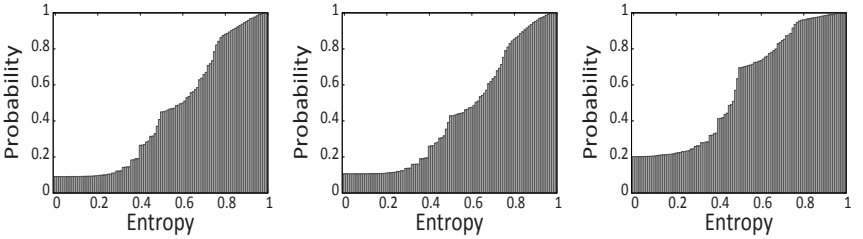
5.3 Experimental Setup

For Enron Mail, we chose 21,262 mails from Enron Mail to be the testing data and extracted 30 days before testing data to be the training data, where the

[1] The Enron Mail data can be downloaded from
`http://www.cs.cmu.edu/~enron/`

[2] The CHT data is not in public, and Chunghwa Telecom's website is
`http://www.cht.com.tw/`

[3] The Reality Mining Dataset can be downloaded from
`http://realitycommons.media.mit.edu/realitymining4.html`



(a) CHT (b) RMD (c) ENRON

Fig. 3. CDF of Time Centrality with 4 time slots in (a) CHT (b) RMD (c) Enron

Table 2. Parameter Settings

Parameter	Meaning	Enron	CHT	RMD
λ	time decay parameter	7 days	3 days	3 days
θ_{out}	outlink weight parameter	5	-	-
ω	time dependency parameter	1	1	1
Time Interval	additional hours next to the current hour	1 hour	1 hour	1 hour

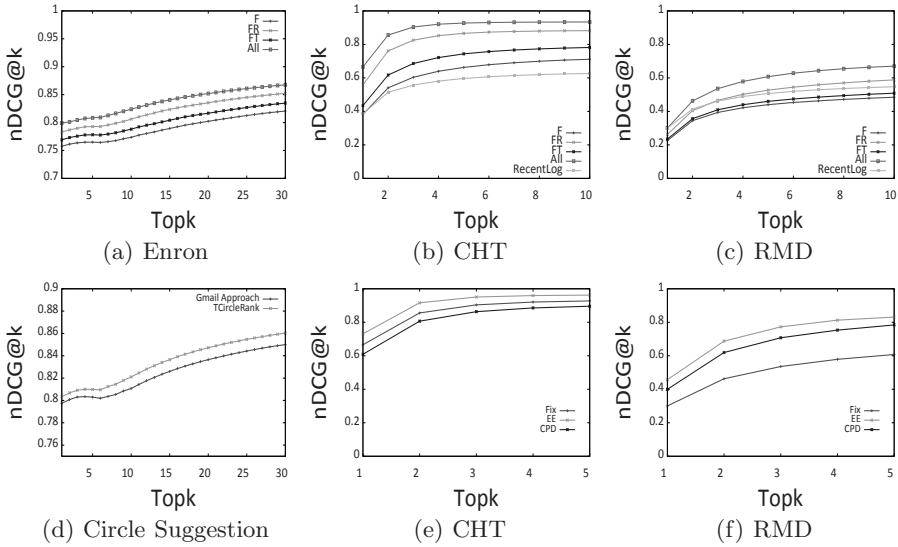
rule in selecting testing data is as follows: (1) the mail should be sent to at least two receivers, or a *group*, and (2) the sender of the mail had sent no less than four mails before. CHT, which is a single interaction data, do not have group information, because CHT consists of cell phone call records and we only need to predict the most likely callee. We chose 30,295 records from CHT to be the testing data and extracted 30 days before testing data to be the training data. The testing data is all in the last day in CHT. We chose 44,166 records from RMD to be the testing data and extracted 30 days before testing data to be the training data. Parameter settings are shown in Table 2.

To evaluate the recommendation quality, we adopt normalized discounted cumulative gain (nDCG) as the measurements. DCG measures the *gain* of a hit result based on its rank in the list, where the top rank has more gain and the lower rank has less gain.

5.4 Circle Recommendation Quality

Evaluation on Seedset Generation: Figure 4(a)(b)(c) shows the impact of each fundamental factor: Frequency (F), Recency (R) and Time-dpendency (T) on Seedset Generation quality. We also compare with RecentLog which directly generates the recommendation list in order by the recent contacts.

In Fig. 4(a), the pink line (All) is our proposal which considers all factors and outperforms other models with at most 4.2% increase in accuracy compare to the baseline. In Fig. 4(b), it is worth mentioning that the lines assemble the log-likelihood, because CHT only has one receiver for recommendation in each record. Our proposal outperforms other models with 26% increase in accuracy compare to the baseline when k is 5. The similar results could be found on



(a) Enron (b) CHT (c) RMD

(d) Circle Suggestion (e) CHT (f) RMD

Fig. 4. nDCG Comparison for Seedset Generation, Circle Suggestion and Time Interval Adjustment

RMD. Fig. 4(c) shows the nDCG comparison for four models and our proposal outperforms other models with 16% increase in accuracy to the baseline when k is 5. Based on above results, we conclude that TCircleRank presents consistent improvement than straightforward suggestion such as frequency or recency.

Evaluation on Circle Suggestion: Fig. 4(d) shows the performance comparison of Gmail Approach [5] and TCircleRank. Because Gmail Approach is a seed-based suggestion approach, we use Seedset Generation to generate a seedset with $k = 3$ and pass the input to Gmail and TCircleRank respectively. The final recommendation list contains the seeds which are different from the original Gmail Approach, but it will not affect the recommendation result because the seeds appears at the top of the list and they are also uncertain receivers for the query user. We use the same test data from Enron Mail as in Fig. 4(a). In Fig. 4(d), the x-axis is top-k ($1 \leq k \leq 30$) and the y-axis is the nDCG value, where the red line is Gmail Approach and the green line is TCircleRank. We can see that no matter in what situation, TCircleRank always has higher nDCG than Gmail Approach.

5.5 Time Interval Adjustment Quality

We compare fixed time interval and our proposed methods. We set δ to 0.5 for Entropy Examination (EE) and τ to 2 for Close Peak Detection (CPD). In Fix Time Interval (Fix), we fixed the time interval to 1 hour because it results in highest nDCG among all fixed time intervals.

Fig. 4(e) shows three methods comparison on CHT. The x-axis is top-k and the y-axis is the nDCG. CPD has the lowest nDCG among three methods, and it

has 4% decrease in accuracy compare to Fix when k is 3. EE outperforms other methods, and it has 5% increase in accuracy compare to Fix when k is 3. On the other hand, Fig. 4(f) shows different comparison results on RMD, where the worst method among the three methods is Fix. CPD becomes an useful method with 17% increase in accuracy compare to Fix when k is 3. This is because that RMD shows stronger time centrality against CHT. EE outperforms the other methods and achieves 24% increase in accuracy compare to Fix when k is 3. We conclude that EE is the best methods for our datasets. In summary, inferring relevant time interval is useful for dynamic circle recommendation.

6 Conclusion

In this paper, we study the problem of suggesting friends by implicit social graph and temporal importance. In this paper, we propose a temporal probabilistic model (TCircleRank) that combine three factors, *Frequency*, *Recency* and *Time-Dependency* to address the fact that *different users have different importance of time for a query user*. Based on TCircleRank, Seedset Generation generates a set of seeds automatically. To recommend circles, we utilize the seedset generated by TCircleRank and considers an additional feature, *Direction* of interactions in our Circle Suggestion approach. We enhance the probabilistic model by further dynamically determine the time interval, which is a parameter to identify time-dependent interactions in derived time intervals. Our experiment results show that TCircleRank and dynamic circle recommendation system are effective on three real datasets, Enron Mail, CHT call detail records and Reality Mining Dataset. We also show that inferring optimal time interval is useful for dynamic circle recommendation. We will extend TCircleRank by automatically deciding the number of seeds and using user clusters. We will further apply our approach in other applications such as content-based sharing and temporal community detection.

References

1. Bartel, J., Dewan, P.: Towards hierarchical email recipient prediction
2. Koren, Y.: Collaborative filtering with temporal dynamics. Communications of the ACM 53(4), 89–97 (2010)
3. Lathia, N., Hailes, S., Capra, L., Amatriain, X.: Temporal diversity in recommender systems. In: Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2010), pp. 210–217 (2010)
4. Liben-Nowell, D., Kleinberg, J.: The link-prediction problem for social networks. Journal of the American Society for Information Science and Technology 58(7), 1019–1031 (2007)
5. Roth, M., Ben-David, A., Deutscher, D., Flysher, G., Horn, I., Leichtberg, A., Leiser, N., Matias, Y., Merom, R.: Suggesting friends using the implicit social graph. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 233–242. ACM (2010)

6. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. Advances in Artificial Intelligence 2009, 4 (2009)
7. Sun, Y., Han, J., Aggarwal, C.C., Chawla, N.V.: When will it happen?: Relationship prediction in heterogeneous information networks. In: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, pp. 663–672. ACM (2012)
8. Wu, S., Sun, J., Tang, J.: Patent partner recommendation in enterprise social networks. In: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, pp. 43–52. ACM (2013)
9. Xiang, L., Yuan, Q., Zhao, S., Chen, L., Zhang, X., Yang, Q., Sun, J.: Temporal recommendation on graphs via long-and short-term preference fusion. In: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 723–732. ACM (2010)
10. Yang, S.-H., Long, B., Smola, A., Sadagopan, N., Zheng, Z., Zha, H.: Like like alike: Joint friendship and interest propagation in social networks. In: Proceedings of the 20th International Conference on World Wide Web, pp. 537–546. ACM (2011)
11. Zheng, N., Li, Q.: A recommender system based on tag and time information for social tagging systems. Expert Systems with Applications 38(4), 4575–4587 (2011)