Predicting User Activity Level in Social Networks

Yin Zhu†, Erheng Zhong†, Sinno Jialin Pan‡, Xiao Wang⋆, Minzhe Zhou⋆, Qiang Yang†⋄
†Hong Kong University of Science and Technology, Hong Kong, China
‡Institute for Infocomm Research, Singapore
⋆Renren Inc., Beijing, China
⋄Huawei Noah's Ark Lab, Hong Kong, China
{yinz,ezhong,qyang}@cse.ust.hk, jspan@i2r.a-star.edu.sg
{xiao.wang,minzhe.zhou}@renren-inc.com

ABSTRACT

Social media such as Facebook, Renren and Twitter provide an ideal ground to study how to predict users' future activities based on their past social behavior. An important measure of the behavior is activity level, such as users' level of *weekly activeness*, or binary classifications in terms of active or inactive. This prediction problem is closely related to Social Customer Relationship Management (Social CRM). Compared to traditional CRM, social CRM exhibit some special characteristics, in terms of user diversity, social influence, and dynamic nature of social networks. Users' social diversity property implies that a global predictive model may not be precise for all users. However, the historical data of individual users are too sparse to enable high-quality personalized models. The social influence property suggests that relationships between users can be embedded to further boost the prediction results on individual users. Finally, the dynamical nature of social networks means that users' behaviors change over time. To address these challenges, we develop a personalized and socially regularized time-decay model for accurate user activity level prediction. We conduct experiments on the social media Renren to validate the effectiveness of our proposed model to demonstrate the superior performance when compared with traditional supervised learning methods as well as node classification methods in social networks.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: [Data mining]

Keywords

Social Network Analysis, User Activity, Prediction

1. INTRODUCTION

The number of active users in a social network is a critical measure of its popularity, which can be used as a signal of investment value for investors. In many social network companies' quarterly reports, such as that of Facebook (FB)[1] and Renren (RENN)[2], the number of Monthly Active Users (MAUs) and other user activeness measures are published. Because these numbers are the strong indicators of popularity and investment value, social network companies adopt various strategies to attract new users and maintain old users, e.g., building faster and more stable services, providing better recommendations, developing innovative UI, and supporting personalized services. An important strategy for increasing the number of active users is to give incentives to users who are inactive or going to be inactive. But if a user is already inactive for a long time (i.e., lost users), it is much harder to activate the user again than when he/she only shows a sign of becoming inactive. This fact motivates us to explore how to predict a user's future activeness either in terms of the level of activity or binary classification into active or inactive. By accurately predicting the future activity levels of users, we can track potentially lost users in an early stage and give them incentives to stay active. The prediction model can also shed lights on the explanation of what user behaviors show correlations with their future activity levels. These insights can help improve user-maintenance strategies.

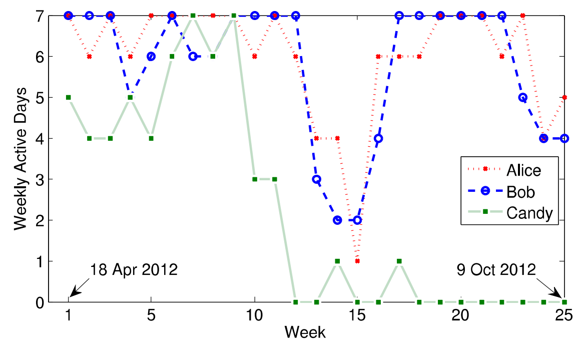

Figure 1: The weekly active days of three Renren users over 25 **weeks.**

Figure 1 shows the weekly online days of three users over 25 weeks from Apr. 18, 2012 to Oct. 9, 2012 in Renren (the details of the dataset are described in the experiment section). Among the three users, Alice and Bob are close friends and exhibit similar patterns of weekly online days. Candy becomes relatively inactive after the 11'th week, and has no more online actions since the 18'th week. If we were able to identify a user like Candy who was still active by some time point but was about to decline his/her online

[1]Reports are available at `http://investor.fb.com`
[2]Reports are available at `http://ir.renren-inc.com`

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2505515.2505518.

activities afterwards, we could try to give them incentives to make them remain active in the social network, e.g., by providing them new services, free e-gifts, gaming points, etc. If we define a *weekly active user* as a user who is online for at least three days during one week, then Alice and Bob are mostly weekly active while Candy is inactive after the 11'th week. Our goal is to make an early prediction in the 11'th week to recognize that Candy would be potentially inactive in the coming week based on her online behaviors during the first 11 weeks.

This problem has also been studied in traditional Customer Relationship Management (CRM) for many years [5, 16]. For example, the three data mining tasks in KDD Cup 2009 organized by Orange [17] were to make predictions on mobile phone users, including switching providers, buying new products, and upgrading services. And in particular, when predicting users who have strong tendencies to leave a service provider, it is called Churn Prediction [10, 21, 22]. These prediction tasks are usually solved by constructing useful features and building a good classifier, or an ensemble of a few classifiers with the features. However, in the context of social networks, new challenges arise. First, in a social network, users may be diverse, and their behavior patterns can be very different among each other. As a result, a global prediction model trained on all users may fail to generalize well on individuals. On the other hand, historical data of individual users are too sparse to train personalized models precisely. Second, the dynamic nature of social network services and individual behaviors cannot be captured well in a static model. As a result, predictive models which contain a factor to model behavior changes are more desirable. Last but not least, in a social network, users are more or less influenced by their friends, and close social friends may exhibit similar active patterns. Therefore, how to leverage the social network structure is critical to make good predictions on user activeness.

Note that the above three challenges are not specific to the activity level prediction problem. One or two of them are common to various prediction tasks. For example, the tasks of email spam detection [4] and email importance ranking [1] share the same problem of personalization, typical time series modeling needs to deal with the dynamics in the temporal space [31], and recently social recommender systems and node classification in social networks usually require to encode social structures into model learning [25, 38]. However, dealing with the three challenges simultaneously makes our problem unique, and we propose a learning model to overcome the three challenges in a unified learning framework.

Our proposed solution starts with a simple model based on logistic regression. We then extend the base model by equipping different terms to capture the three properties of the user activity level prediction task, resulting in a unified optimization problem. Specifically, to address the user diversity issue, we propose to decompose the model into two parts: the common part, which is for global optimization over all users, and the user-specific part, which is for personalized optimization on specific users. However, as described above, historical data of each individual user are extremely sparse to optimize personalized models independently. Therefore, we propose to jointly learn personalized models for individual users by making use of the common part of the models as a bridge. To model the dynamics in user behavior, we introduce a time-decay term to penalize out-of-date training data. To model social influence, we deploy a social regularization term for smooth predictions over close friends and groups of users whose activity levels are also close to each other.

In summary, the main contributions of this paper include:

Table 1: Definition of Notations

Notation	Notation Description
	Data
G	Social network
N	Number of users
T	Number of time periods
$\mathrm{x}_i^{(t)}$	Feature vector of the i-th user in period t
$y_i^{(t)} \in \{+1, -1\}$	Activeness status (inactive v.s. active)
p	Number of features
$S^{(t)}$	Users' social tie matrix during time period t
	Model
w_0	Common model coefficients
w_i	Model coefficients of the i-th user
γ, γ_0, β	Regularization parameters
α	Time-decay parameters

- We propose a learning model that encodes users' personalization, social influence and dynamic behaviors into a unified optimization framework for the user activity level prediction task.
- We validate the effectiveness of our proposed model by comparing several baselines on a large-scale realworld social media, Renren. Furthermore, for this specific social media, we constructed three categories of useful features and show their effectiveness in activity level prediction.

The rest of the paper is organized as follows: In Section 2, we define the activity level prediction task as a classification problem and summarize the notations used through the paper. In Section 3, we start with introducing a base logistic regression model, then propose a unified framework to encode three components to capture the three properties of the task, and finally show how to optimize the unified model. After that we conduct extensive experiments in Section 4. We finally review some related works and conclude this paper in Sections 5 and 6 respectively.

2. PROBLEM FORMULATION

We formally define the problem of user activity level prediction in this section. The definition of notations can be found in Table 1. In a social network G, suppose there are N users, and each user i in the time period t can be represented by a p-dimension feature vector, $\mathrm{x}_i^{(t)} \in \mathbb{R}^p$. This feature vector represents various user activity information till the time period t. The corresponding label of $\mathrm{x}_i^{(t)}$, denoted by $y_i^{(t)} \in \{+1, -1\}$, is the activeness status (inactive v.s. active) in the next period $t+1$. The goal is to learn a prediction function $f(\cdot)$ that takes the features $x_i^{(t)}$ of user i in the time period t as inputs and predict his/her activeness status y_i in the next time period $t+1$.

We use a matrix $S^{(t)}$ to represent the social ties between users during the time period t. If user i and user j are not friends during the time period t, then $S_{ij}^{(t)} = 0$, otherwise, $S_{ij}^{(t)}$ is defined as follows,

$$S_{ij}^{(t)} = \frac{I(i,j)}{\sum_{k \in \mathcal{N}(i)} I(i,k)}, \quad (1)$$

where $I(i,j)$ denotes the number of social interactions (e.g., leaving messages, visiting homepages, etc.) between user i and user j, and $\mathcal{N}(i)$ denotes the set of friends of user i. Here we assume that a larger value of $S_{ij}^{(t)}$ suggests a stronger friendship between the users i and j in time period t.

3. PERSONALIZED TIME-DECAY LOGISTIC REGRESSION WITH SOCIAL REGULARIZATION

As stated in the previous section, we formulate the activity level prediction task as a binary classification problem. In this section, we propose a model to address it based on logistic regression. Generally, logistic regression builds a linear function on input features, and predicts target labels using the sigmoid function as follows,

$$\hat{y}_i = \sigma(\mathrm{w}^\top \mathrm{x}_i) = \frac{1}{1 + \exp(-\mathrm{w}^\top \mathrm{x}_i)},$$

where x_i is a data instance, $\hat{y}_i$ is the corresponding prediction, and w is the coefficient vector to be learned. By using logistic regression, a base model for activity level prediction can be formulated as learning an optimal solution w_0 by solving the following minimization problem,

$$\min_{\mathrm{w}_0} \mathcal{J} = \sum_{i=1}^{N} \sum_{t=1}^{T} \ell\left(y_i^{(t)}, \mathrm{w}_0^\top \mathrm{x}_i^{(t)}\right) + \gamma_0 \|\mathrm{w}_0\|_2^2 \tag{2}$$

where T is the number of time periods observed so far, γ_0 is a parameter on the regularization term $\|\mathrm{w}_0\|_2^2$ penalizing the model complexity, and the loss function $\ell(y_i, \mathrm{w}_0^\top \mathrm{x}_i)$ is defined as

$$\ell\left(y_i^{(t)}, \mathrm{w}_0^\top \mathrm{x}_i^{(t)}\right) = \log\left(1 + \exp\left(-y_i^{(t)} \mathrm{w}_0^\top \mathrm{x}_i^{(t)}\right)\right). \tag{3}$$

An advantage of using logistic regression as the base classifier is that it can generate a probabilistic output of a user being inactive or active. This is important in real-world applications as users can be ranked according to their probabilities being active or inactive such that different marketing strategies can be adopted based on the ranking. However, this base model fails to embed the three important properties of the activity level prediction task in learning: user diversity, dynamic behaviors and social influence. As stated in Section 1, different users may have very different activities, and their activities may be further influenced by their friends. Furthermore, users' activities can change over time. Therefore, in the following, we show how to extend the base logistic regression model to capture these three specific characteristics.

3.1 Personalization

As mentioned, different users may have different behaviors, and the correlations between users' behaviors and their activity levels may be different. For example, some users may prefer to leave messages on their friends' homepages while others may prefer to talk with them directly using web chat. In addition, for those who log in a social media system frequently, some may only visit their friends' homepages, but others may have lots of interactions with their friends. This implies that for different users, the prediction models on activity levels should be different. Therefore, using a global model (w_0 in particular) learned in (2) cannot make precise predictions on all users. Inspired by multi-task learning [15], which aims to learn a set of different but related tasks jointly by exploring the commonality across tasks, we consider building a personalized model for an individual user as a task. We further assume that for each task, the predictive model can be decomposed into two parts. One is referred to as a common part shared by multiple tasks and the other is referred to as a specific part for individual tasks. Therefore, we introduce a user-specific parameter w_i for each user, and plug it into (2) as follows,

$$\begin{aligned}\min_{\mathrm{w}_0, \{\mathrm{w}_i\}_{i=1}^N} \mathcal{J} = & \sum_{i=1}^{N} \sum_{t=1}^{T} \ell\left(y_i^{(t)}, (\mathrm{w}_0 + \mathrm{w}_i)^\top \mathrm{x}_i^{(t)}\right) \\ & + \gamma_0 \|\mathrm{w}_0\|_2^2 + \gamma \sum_{i=1}^{N} \|\mathrm{w}_i\|_2^2, \end{aligned} \tag{4}$$

where γ is a parameter on the regularization terms for individual users. Note that the global knowledge across users can be modeled through the global parameter w_0 and users' specific patterns can be captured by the user-specific parameters $\{\mathrm{w}_i\}$'s. The tradeoff between commonality across users and extreme personalization lies in the ratio between the values of γ_0 and γ. For instace, we can set γ a relatively small value to allow more personalization, and this however may cause overfiting to individual users' training data.

3.2 Dynamical Modeling

Another important property in user activity level prediction is the dynamic nature of users' behaviors over time. For example, a user may interact with his/her friends frequently when he/she is still a student, but may only visit his/her friends' homepages after graduation due to the lack of time. Another example is that, a user may be active when a new application or game is launched on a social media, but may become inactive when he/she looses interests in it. In summary, a user's activity level may be similar in short time but may become more and more different in long time. Therefore, motivated by the network dynamic model [31], we further introduce a term, $e^{-\alpha(T-t)}$, to model time decay into the personalized model in (4) to reduce the impacts of out-of-date training data,

$$\begin{aligned}\min_{\mathrm{w}_0, \{\mathrm{w}_i\}_{i=1}^N} \mathcal{J} = & \sum_{i=1}^{N} \sum_{t=1}^{T} e^{-\alpha(T-t)} \ell\left(y_i^{(t)}, (\mathrm{w}_0 + \mathrm{w}_i)^\top \mathrm{x}_i^{(t)}\right) \\ & + \gamma_0 \|\mathrm{w}_0\|_2^2 + \gamma \sum_{i=1}^{N} \|\mathrm{w}_i\|_2^2, \end{aligned} \tag{5}$$

where α is a parameter to control the decay rate. Note that the weights of the training data decrease exponentially with time increases. Based on the objective in (5), more recent training data play more important roles in model learning.

3.3 Social Regularization

The major difference between social and traditional user activity level prediction is social influence. In social networks, users usually interact with their friends, and thus their activity levels tend to be influenced by their friends', and vice versa. Intuitively, one may be active if a few of his/her friends are active, while may become inactive if most of his/her friends are inactive. Furthermore, the impact of the social influence may depend on the strength of the relationships, as users may only be influenced by their close friends. Formally, we introduce the following regularization term to smooth the prediction results so that the prediction of a user's activity level is similar to that of his/her *close friends*,

$$\sum_{i=1}^{N} \sum_{t=1}^{T} \sum_{j \in C_i^{(t)}} \left(\mathrm{w}_i^\top \mathrm{x}_i^{(t)} - \mathrm{w}_j^\top \mathrm{x}_j^{(t)}\right)^2 \tag{6}$$

where $C_i^{(t)}$ is the set of x_i's close friends based on social interaction counts in the t-th time period. How to choose the subset C_i from the full list of x_i's friends is based on (1), which will be discussed in experiments. By adding this regularization term, knowledge in users' social relations can be encoded into the model. We notice that, the social regularization term is only performed on users' close friends, which is consistent with real-world applications, as most users are influenced by only a few close friends. This also brings two advantages: 1) it accelerates the model computation as less data are considered, and 2) it boosts the model performance in activity level prediction as most irrelevant data are eliminated, which will be verified in experiments.

3.4 Overall Optimization Problem

We now embed all components described in Section 3 into a unified optimization problem for user activity level prediction as follows,

$$
\begin{aligned}
\min_{\mathrm{w}_0, \{\mathrm{w}_i\}_{i=1}^N} \mathcal{J} = \quad & \sum_{i=1}^{N}\sum_{t=1}^{T} e^{-\alpha(T-t)} \ell\left(y_i^{(t)}, (\mathrm{w}_0 + \mathrm{w}_i)^\top \mathrm{x}_i^{(t)}\right) \\
& + \beta \sum_{i=1}^{N}\sum_{t=1}^{T} \sum_{j \in C_i^{(t)}} \left(\mathrm{w}_i^\top \mathrm{x}_i^{(t)} - \mathrm{w}_j^\top \mathrm{x}_j^{(t)}\right)^2 \\
& + \gamma_0 \|\mathrm{w}_0\|_2^2 + \gamma \sum_{j=1}^{N} \|\mathrm{w}_j\|_2^2. \qquad (7)
\end{aligned}
$$

We call this model *Personalized Time-Decay Logistic Regression with Social Regularization* (**SocTiPerLR**). By learning model coefficients with multiple regularization terms collectively, personalization, social ties and dynamical knowledge can be encoded to build a more accurate model. To learn the optimal solutions of w_0 and $\{\mathrm{w}_i\}_{i=1}^N$, we propose to use gradient descent methods. It can be shown that the derivatives of the objective $\mathcal{J}$ with respect to w_0 and each w_i can be computed as

$$
\begin{aligned}
\frac{\partial \mathcal{J}}{\partial w_0} = \quad & \sum_{i=1}^{N}\sum_{t=1}^{T} e^{-\alpha(T-t)} x_i^{(t)} \big(y_i^{(t)} - \sigma(\mathrm{w}_0^\top \mathrm{x}_i^{(t)})\big) + \gamma_0 w_0 \\
\frac{\partial \mathcal{J}}{\partial w_i} = \quad & \sum_{t=1}^{T} e^{-\alpha(T-t)} x_i^{(t)} \big(y_i^{(t)} - \sigma(\mathrm{w}_i^\top \mathrm{x}_i^{(t)})\big) \\
& + \beta \sum_{j \in C_i} \left(\mathrm{w}_i^\top \mathrm{x}_i^{(t)} - \mathrm{w}_j^\top \mathrm{x}_j^{(t)}\right) \mathrm{x}_i^{(t)} + \gamma w_i,
\end{aligned}
$$

Based on the above derivatives, we can update w_0 and w_i alternatively by using the following rules till the solutions are converged,

$$
\mathrm{w}_0 \leftarrow \mathrm{w}_0 - \eta \frac{\partial \mathcal{J}}{\partial \mathrm{w}_0}, \qquad (8)
$$

$$
\mathrm{w}_i \leftarrow \mathrm{w}_i - \eta \frac{\partial \mathcal{J}}{\partial \mathrm{w}_i}, \qquad (9)
$$

where η is the learning rate. That is, in each iteration, we first fix all $w_i (i = 1, \ldots, n)$ and optimize w_0, and then we fix w_0 and optimization w_i. The overall algorithm for **SocTiPerLR** is summarized in Algorithm 1.

3.5 Computational Analysis

The computational cost in each iteration (Steps 5-8 in Algorithm 1) is approximately the cost of running two plain logistic regressions [27] on the whole dataset. In our experiments, we show the convergence speed empirically.

Notice that steps 6-8 in Algorithm 1 can be massively parallelized because the N user specific models can be trained independently. The speedup factor of such data parallel tasks [34] is proportional to the number of CPU cores the program is given. Further more, the optimization of the global model (Step 5) can

Algorithm 1 Gradient Decedent Optimization for SocTiPerLR

1: **Input**: user features $\mathbf{X} = \{\mathbf{X}^{(t)} = \{\mathrm{x}_i^{(t)}\}_{i=1}^N\}_{t=1}^T$, labeled data Y, regularization parameters γ, learning rate η and maximal number of iterations I
2: **Output**: Common model w_0 and specific models for all users ${\mathrm{w}_i}_{i=1}^N$
3: Generate w_0 and all w_i randomly
4: **for** $i = 1$ to I **do**
5: Fix other parameters, and keep updating $\mathrm{w}_0 \leftarrow \mathrm{w}_0 + \eta \frac{\partial \mathcal{J}}{\partial \mathrm{w}_0}$
6: **for** $n = 1$ to N **do**
7: Fix other parameters, and keep updating $\mathrm{w}_i \leftarrow \mathrm{w}_i + \eta \frac{\partial \mathcal{J}}{\partial \mathrm{w}_0}$
8: **end for**
9: **IF** convergence **break**
10: **end for**
11: **Return** w_0 and ${\mathrm{w}_i}_{i=1}^N$

also be done in parallel using techniques presented in [24]. Therefore, the implementation of Algorithm 1 is fast and can scale up to distributed environments.

4. EXPERIMENTS

In this section, we compare the proposed activity level prediction method with several baselines on a real-world data set. Through extensive experimental results, we demonstrate the effectiveness of our model, and the impacts of the three components in our model: user personalization, modeling dynamic behaviors and social regularization. Furthermore, we also report the features designed for social activity level prediction in detail.

4.1 Data Description

The data set used for evaluation is collected from a real-world social media, Renren.com, which is one of the largest online social media in China and has over 170 million registered users. Similar to Facebook, Renren is an undirected friendship network with a mature application platform to support various social services. On the platform, users can perform various actions such as social messaging and gaming. To prepare the evaluation data set, we extract a subnetwork from the whole social network in Renren by applying a community detection algorithm [7]. The subnetwork contains $26,418$ users in total after removing those who have no activity during 18 April 2012 and 9 October 2012 (25 weeks in total). A user in the social network can perform many actions, from updating statues, sending messages, posting photos, to playing various social games. Their activity levels are expressed by these actions. We extract the action log between the 25 weeks for these users. The user action log, which is summarized from the raw HTTP requests, is a content-less [3] log of users' online activities, including updating statues, posting/replying on walls, commenting on photos, checking notifications, etc. These actions can be classified into two categories. The first category of actions does not involve other users, e.g., uploading a photo, posting a message, etc. While the other category of actions, which constitute most of the actions of a user, are interactions with other users, e.g., a message is sent from user a to user b, user a's photos are viewed by user b. Actions in the latter category are used to calculate the strength of friendship between user i and user j during week t, $S_{i,j}^{(t)}$ (Eq. 1).

4.2 Feature Construction

[3] Though all the user names have been anonymized, to protect the user privacy, we avoid using any actual content (e.g. the actual message content).

Table 2: Summary of Features

Category	*Feature description for* $\mathrm{x}_i^{(t)}$
Action features	Number of status posted Number of photos posted Number of searches Number of friend applications Number of accepted friend applications Number of denied applications Number of replied messages Number of likes Number of videos played Number of songs played Number of blogs visited Number of photos visited Number of notification checks Number of forwarded statuses ... (more actions)
Time series features	Number of active days of week t Mean number of active days over the weeks in the window Standard variance of the number of active days over the weeks Ratio of active weeks in the window Mean number of days among the active weeks If the last week is active, count backwardly until an inactive week occures If the last week is inactive, count backwardly until an active week occures
Social features	Number of social friends Number of active social friends in week t Number of social friends with interactions in week t

Remind that for a user i in the t'th week, we need to generate a feature vector $\mathrm{x}_i^{(t)}$ to represent it. In practice, it has been proven that a lot of machine learning tasks resort to feature engineering rather than complex modeling to boost classification accuracies [12]. For example, feature engineering has proven to be effective in recent data mining competitions [20, 39], and social churn prediction [21]. In this section, before conducting experiments to verify our proposed model, we first introduce the features designed for the user activity level prediction task on social media. For a user i in the t'th week, the feature vector $\mathrm{x}_i^{(t)}$ consists of three different groups of features as follows, which are also summarized in Table 2.

1. A first group of features is referred to as action features. From the action log, we use 31 frequent actions and count the times of every action that user i has performed during the t'th week. We have also calculated different versions of the counts conditioned on time, e.g. splitting time into Weekdays or Weekend, and splitting time into Morning, Afternoon, Evening or Night. These conditioned features have proven to be useful in sensor-based activity recognition [39].

2. A second group of features is referred to as time series features on activity level. For the t'th week, we take the past k weeks into consideration and construct an active-or-not series of length k. We then extract some statistics from the series as our features. These statistics features include the average length of continuous active weeks, last inactive week, the ratio between active weeks and inactive weeks, etc. The parameter k is set to 5 and 10 to generate two versions of the features. Note that a similar set of features has been used for detecting Internet path changing [8].

Table 3: Sample Statistics for Evaluation Weeks

Label	*#21*	*#22*	*#23*	*#24*	*#25*
active→inactive	1732	1733	1709	1846	2112
active→active	13418	13476	13291	13088	12481

3. A third group of features is referred to as social features. We have also extracted several features from the structure of the social network among users, which have proven to be useful in churn prediction [22, 28]. Such features include number of social friends of user i, number of social friends with interactions to user i during week k, number of social friends who are active during week t. For the later two features, we also calculate the normalized versions, i.e., dividing them by the number of social friends of user i.

4.3 Evaluation Methods

We hold out the data from the 21'th week to the 25'th week for testing. Specifically, we use the first 20 weeks of data for training and the subsequent week after the 20'th week for testing. For example, to evaluate the predictions on week 21, we use data from the 1'th week to the 20'th week to build the model, and apply the model on the feature vectors of the 21'th week, $\{\mathrm{x}_i^{(21)}\}_{i=1}^n$. The data sample counts for the five testing weeks are shown in Table 3. As mentioned in Section 1, if a user has online actions for at least three day during week t, we label him/her as active for week t; otherwise, inactive. We observe that the distribution is unbalanced over the two classes and thus we need to adjust the ratio between the weights of active and inactive instances [11]. This can be simply done by adding different weights or cost on different classes to the loss function in (3). We denote by b_{active} and $b_{inactive}$ the weights on the two classes *active* and *inactive* respectively.

We compare the proposed algorithm, Personalized Time-Decay Logistic Regression with Social Regularization (**SocTiPerLR**), with two types of baselines:

1. Logistic Regression (**LR**) and RandomForest (**RF**). These two classifiers, one of which is linear and the other is non-linear, are commonly used in various classification tasks. Using these two classifiers we can first explore the discriminative power of the proposed features presented in Table 2, and further verify how much improvement our proposed model can bring.

2. A state-of-the-art node classification algorithm (**Node**) proposed by [38]. In this algorithm, a user can be only represented by one feature vector instead of time-series feature vectors. We thus choose the latest feature vector to represent each user. This model considers the similarity among friends, but still uses a single global model to make predictions on all users.

Precision, recall and F1-score are often used as evaluation criteria in churn prediction tasks [21, 22, 28]. Following this common practice, we use them as evaluation criteria in this paper. These measures are defined for the active-to-inactive users, i.e., users whose activity level declines from active to inactive in the coming week:

$$\begin{aligned}\text{recall} &= \frac{\#\text{correctly classified inactive users}}{\#\text{true inactive users}}\\ \text{precision} &= \frac{\#\text{correctly classified inactive users}}{\#\text{users classified as inactive}}\\ \text{F1-score} &= \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}\end{aligned}$$

We tune the parameters of all models by considering users' activeness in the 20'th week as targets and generating feature vectors from data observed in the first 19 weeks, and fix the parameters in all experiments for making predictions on the weeks 21 to 25.

4.4 Strong Social Ties Construction

Social regularization is a critical component in the proposed model, and the question is which part of a users' friends are supposed to be similar to him/her with respect to weekly online days. A previous research on Facebook's social interaction [36] shows that most of a user's social interactions are played with only a few close friends. We verify this finding in Renren network as shown in Figure 2. Each curve shows a Cumulative Distribution Function (CDF) of an interaction distribution. For instance, the solid line shows the proportion of the interactions of the top friend over all friends of a user. From this CDF curve, we observe that about 10% of users have more than 50% of interactions with a single friend. To draw the further conclusion that a user's activity level is closely related with the friends with whom this user interacts most, we plot the distribution of the following ratio:

$$r_i^{(t)} = \frac{d_i^{(t)}}{1/|C_i^{(t)}|\sum_{j\in C_i^{(t)}} d_j^{(t)}}, \quad i = 1, 2, \ldots, N, \qquad (10)$$

where $d_i^{(t)}$ is the number of user i's active days during week $t-3$ to week t (to obtain a stable statistics, we use one month rather than one week.) We use different methods to construct $C_i^{(t)}$: 1) friends with whom the user has at least 3 interactions and the ratio accounts at least 15% of his total social interactions (i.e. for user i, select all $\{j\}$'s with $S_{ij}^{(t)} \geq 15\%$ as the close friend $C_i^{(t)}$); 2) all friends of user i (i.e. $C_i^{(t)} = \mathcal{N}(i)$); and 3) a random set of users with a size equal to the number of user i's friends. We plot the distribution of $r_i^{(4)}$ in Figure 8, we can find that the ratio distribution is centered for close friends; however the distribution for the whole friends is far from that of the close friends and is even close to that of random users. This fact shows that only knowledge from close friend may be helpful to regularize the model building. The two peaks in the density are due to that the online days of the users are large at the two extremes, i.e., during a week, more users are active for 7 or 0 days than for 3/4 days.

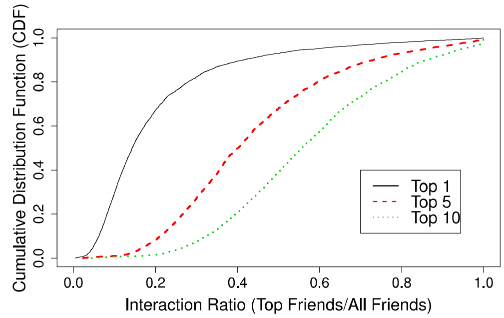

Figure 2: Interaction Distribution for Top 1/5/10 Friends.

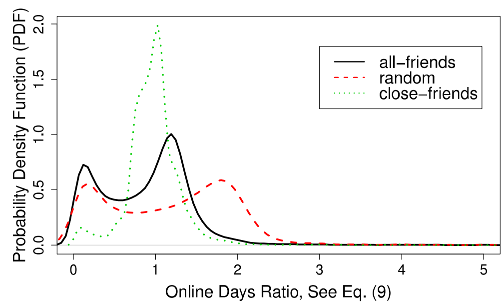

Figure 3: Ratio Distribution for Online Days.

4.5 Performance Comparison

The performance of the model proposed in this paper and the baselines are presented in Table 4. We can find that **LR** which considers only users' features, performs the worst. By accounting the non-linear prediction ability, **RF** performs slightly better than **LR** but still worse than our method, as **RF** does not take three important factors into account. **Node** can only take one sample from each user, and though it uses social regularization it does not model user variety and time decay. The improvement of our proposed model **SocTiPerLR** over the three baselines is obvious. We also show the precision and recall in Tables 5 and 6. Take Week 21 as an instance, the 11.8% recall improvement over **LR** means that our model can find 204 more active→inactive users out of the total 1732. Through the comparison, we can also find that though we have three features that encode the social information, the two baselines **LR** and **RF** cannot use them to reach the performance of our proposed method.

Table 4: Performance Comparison of Different Methods in terms of F1-score

Method	*#21*	*#22*	*#23*	*#24*	*#25*
LR	0.512	0.498	0.521	0.492	0.523
RF	0.529	0.507	0.525	0.495	0.533
Node	0.510	0.503	0.516	0.489	0.531
SocTiPerLR	0.583	0.553	0.551	0.542	0.561

Table 5: Performance Comparison of Different Methods in terms of Precision

Method	*#21*	*#22*	*#23*	*#24*	*#25*
LR	0.403	0.392	0.411	0.379	0.533
RF	0.422	0.398	0.414	0.386	0.430
Node	0.403	0.394	0.417	0.377	0.429
SocTiPerLR	0.452	0.441	0.431	0.419	0.436

Table 6: Performance Comparison of Different Methods in terms of Recall

Method	*#21*	*#22*	*#23*	*#24*	*#25*
LR	0.701	0.681	0.712	0.700	0.685
RF	0.708	0.699	0.716	0.689	0.701
Node	0.693	0.695	0.675	0.693	0.697
SocTiPerLR	0.819	0.742	0.761	0.768	0.786

Table 7: Performance Comparison on Different Components in terms of F1-score

Method	*#21*	*#22*	*#23*	*#24*	*#25*
PerLR	0.553	0.533	0.542	0.533	0.547
TiPerLR	0.563	0.542	0.545	0.535	0.550
SocTiPerLR	0.583	0.553	0.551	0.542	0.561

Our next experiment is to compare two reductions of **SocTiPerLR**: **PerLR** which only considers the personalization factor in (4) and **TiPerLR** which has personalization and time decay but ignores the social factor in Eq. (5). The comparison results are showed in Table 7. The improvement of **PerLR** over the basic logistic regression (**LR** in Table 4) is significant , as high as 0.04 on F1-score in all comparisons. This confirms our observations that the users are different in terms of their behaviors and feature description. Due to the limited number of instances for each users ($\leq$ 20) and the short time range (20 weeks), decaying the importance of the training instances temporally may not bring significant improvements. However, adding time decay to the personalized logistic regression also outperforms PerLR by 0.01 on F1-score in the 21'st week. The proposed method **SocTiPerLR** performs best among all the algorithms over all weeks and improves the F1-score of TiPerLR by 0.01 on average. This, from the empirical aspect, supports the necessity to consider three important factors.

4.6 Sensitivity Analysis

In the following we study how the model parameters affect the performance of **SocTiPerLR** and look in depth the contribution of each component in our model. To study the effort of the class

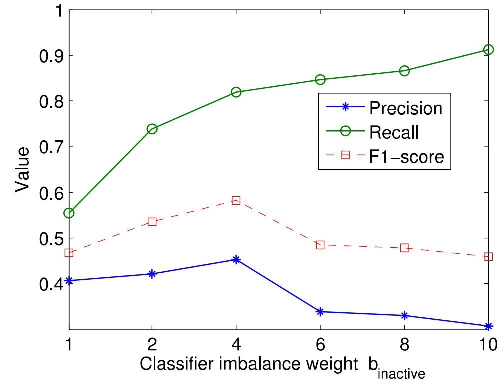

Figure 4: Change the label weight between two classes.

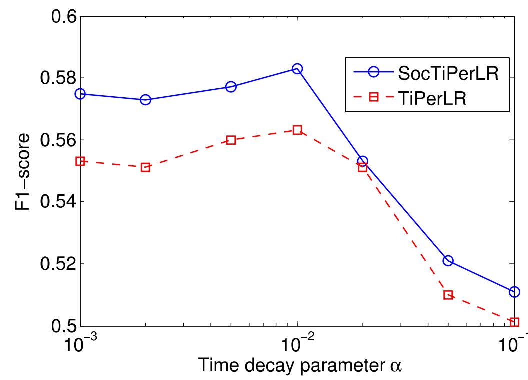

Figure 5: Change the time decay parameter α.

weight on the recall/precesion/F1-score, we set the weight of active samples (b_{active}) as 1 and change the weight for inactive samples (b_{inactive}) from 1 to 10, and show the precision/recall/F1-score trend in Figure 4 . We can see our model generally obtains a good recall though the precision is not high for all weights. When the weight for inactive samples is 4, the model has the best F1-score. Considering that among the users who are active this week, and only a fraction of them will fall inactive next week and that the online user behavior is also heavily affected by their offline activity which is beyond our knowledge, the 0.78 recall is actually not bad, while the precision is also acceptable.

Next we study the time decay parameter α, social regularization parameter β and the personalized regularization parameter γ. While changing one parameter, we fix all other parameters to see how changing the value affects the F1-score of the prediction on week 21. Figure 5 shows the performance of two models **TiPerLR** and **SocTiPerLR** on week 21. We find that a small decay parameter, $\alpha = 0.01$ (the weight for the first week samples is 0.82), works better than large ones. This is because when α is large, the penalization of previous samples is too much and the 20 training samples are too few to provide enough information. Figure 6 shows how the social regularization coefficient β affects the model performance. We can find that when β is too large the regularization can hurt the performance and the F1-score can drop below that of **TiPerLR**. To study the effectiveness of personalized models, we fix the global regularization parameter $\gamma_0 = 1$ and vary the parameter of λ from 0.01 to 100, the performance trend of three models

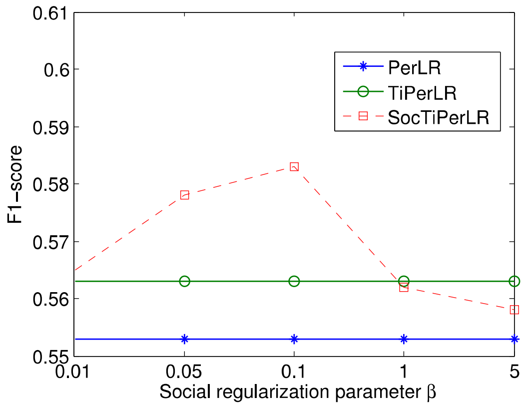

Figure 6: Change the social regularization parameter β.

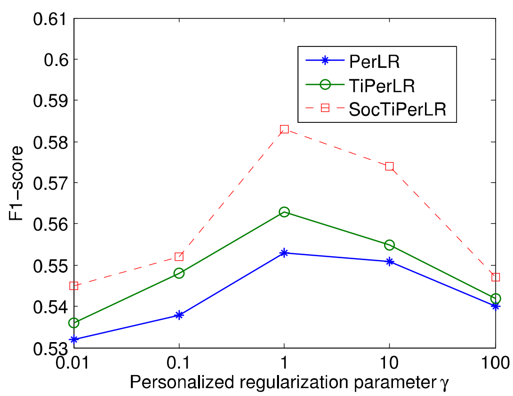

Figure 7: Change the personalized regularization parameter γ.

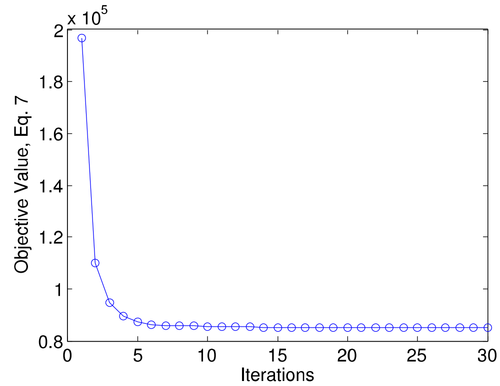

Figure 8: The convergence speed of Algorithm 1.

Table 8: Social Ties Construction Using Different Thresholds

Threshod	*Precision*	*Recall*	*F1-score*
50%	0.433	0.772	0.554
30%	0.441	0.773	0.561
15%	0.452	0.819	0.583
10%	0.437	0.801	0.566
5%	0.411	0.755	0.532

on week 21 is shown in Figure 7. As we have shown in the previous subsection, the choice of the close-friend set C_i may be critical to the performance of the social regularization. For user i, we use different thresholds to select his close friends by filtering $S_{ij}^{(t)}$ using the threshold. Table 8 shows the Precision/Recall/F1-score for different thresholds from 5% to 50% when other parameters are fixed. We find that when using top 15% close friends for social regularization, the perform is best.

The iterative optimization of **SocTiPerLR** generally takes less than 15 iterations to converge, Figure 8 shows the convergence curve when training the model using data from weeks 1 to 20. The second point in the figure shows the objective value after the first iteration. The gap between this value and the final converged objective value verifies that personalization helps to reduce the training error significantly than a global model.

4.7 Discussions

Our experiments show that predicting users' future activity levels in a social network is generally quite hard to have high precision and high recall at the same time. This is not uncommon in activity level prediction in other applications, e.g. churn prediction in online chat rooms [28] and online games [22]. In practice, this precision and recall are already good for social CRM purposes in Renren network. Using our prediction results, we can reduce the number of update emails significantly because for users who are predicted as active-in-the-future we do not need to send them update emails. For such users, it is much better to let them check the updates on the social network. The hardness in accurate prediction is partially due to that the user activity level is also affected by users' offline schedules and cannot be predicted using their online traces alone. How to incorporate the offline behavior to predict the online activity level can be an interesting future work. In this paper, by modeling the three aspects (personalization, social reutilization, and time decay) into a unified model, we archive much better results than a single prediction model, which is a common practice in some previous works [13, 21, 22].

The following table shows the top five features ranked by Random Forest classifier [4]. The importance has been normalized by that of the most discriminative feature. From this table, we can find that two actions, notification check and photo uploading, are high indictors of activeness in the future. This confirms our observations of some existing functionalities in Renren, e.g. a recent feature, today's photo in history [5], brings quite a lot of inactive users to become active again. The user will keep active in the coming week to see whether his/her photos, especially the newly uploaded ones, are being visited and commented. The fact that notification checks have high indication of user activeness can also lead us to improve the current notification check user experience.

5. RELATED WORK

Social CRM, and Social Activity Analysis. The main goal of Social CRM is to keep active users in a social network [16]. This is a very broad research topic, and researchers and practitioners use quite different approaches, e.g., innovative UI [32], social

[4]We use `varImpPlot` command in randomForest R package to perform feature ranking; the variable importance measure is Mean Decrease Gini Index.

[5]On the right bar of the user home page of Renren, it displays photos that were uploaded on the same date years before.

Table 9: Feature Importance

Feature	*Relative Importance*
Number of notification checks	100
Number of active days of week t	77
Ratio of active weeks in the window	69
Number of active social friends	53
Number of photos posted	41

games [2], personalized news/notifications [9], and offline promotion [33]. Our approach is more focused on predicting users who have a tendency to decline their activity levels. Previous literature calls this problem as churn prediction. Social user behavior has been studied recently, e.g., analysis on the user interactions in Facebook [36, 35], activity recommendation [23] and user activity level [3]. [3] is working on similar data as our work, however it mainly focuses on the statistics and analysis, rather than prediction.

Churn Prediction. Churn prediction aims to find users who will leave a network or a service, and by finding such users the service provider can analyze the reason and figure out the strategies to maintain such users. Our problem of activity level prediction is similar to social churn prediction in the aspect that both aim to predict the future activity level of a user. In the past, social churn prediction has been studied in many different application areas, including telecommunications [10, 17, 30], online social games [22], and QA forums [13, 14, 37]. The users in these applications usually do not have complicated user behaviors, e.g., in telecommunications, there are only two main activities, short messaging and calls. Our work enriches the application area by performing analysis to a real-name social network with complicated user behaviors. Most of these research projects fit in the feature engineering theme – encoding user behavior log and social structures as features and building a classifier using these features. The unified model proposed in this paper works in a perpendicular direction by considering the similarity and difference among users. In particular, we allow each user to have a unique prediction model [19], and to improve the generalization ability of each model, we require all the models to have a common part and the user unique part is also regularized by strong social ties.

Modeling techniques. The user difference has been studied previously in Multi-task Learning [15] and Transfer Learning [29]. Our personalized logistic regression model can be viewed as a special multi-task learning model. A similar personalized model is used in Gmail importance ranking system [1]. Model personalization is also used in ad prediction recently [6]. Time decay is a typical modeling technique used in time-series [18]. Our use of social regularization is inspired by the recent advances in personalization, especially in Collaborative Filtering [25]. Although the three main techniques are not new, we take a simple yet practical approach to unify them into the logistic regression classifier and our final model is still quite easy to implement. The computational complexity of our model is also acceptable as it is only a small multiplier (the number of iterations) of that of a single logistic regression.

6. CONCLUSION AND FUTURE WORK

This study is motivated by the need to boost more daily active users in a social network. In this paper, we have studied users' activity level prediction, which is an important task in Social Customer Relational Management (Social CRM). Different from the traditional CRM, social CRM has some special properties: user diversity, dynamic behaviors and social influence. These issues make the problem more challenging. By taking these factors into account, we have designed a unified learning framework that can predict the future activity level of a user in the social network more accurately than baseline methods. These activity predictions not only reduce the cost for user maintenance, but also avoid disturbing normal users, e.g., by sending notification and update emails only to users who have the tendency to drop the activity level in Renren.

For our future work, we will test our models on different social networks other than Renren. Another important future work is to use more information, not only the online user behavior but also the offline user activities. In the past, such offline information was hard to obtain; nowadays social networks have mobile apps for user to use with their smartphones and such apps can sense the physical activities (provided that the user has granted the right). By using the extra information in the offline/physical world and exploiting the online and offline knowledge collectively, the online activity level can be predicted more precisely. This future work can be put in a general research theme, learning from both physical world and virtual world.

7. ACKNOWLEDGMENTS

We thank the support of Hong Kong RGC GRF Projects 621211 and 620812.

8. REFERENCES

[1] D. Aberdeen, O. Pacovsky, and A. Slater. The learning behind the gmail priority inbox. In *NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*, 2010.

[2] G. Abramson and M. Kuperman. Social games in a social network. *Physical Review E*, 63(3):030901, 2001.

[3] F. Benevenuto, T. Rodrigues, M. Cha, and V. A. F. Almeida. Characterizing user behavior in online social networks. In *Internet Measurement Conference*, pages 49–62, 2009.

[4] E. Blanzieri and A. Bryl. A survey of learning-based techniques of email spam filtering. *Artif. Intell. Rev.*, 29(1):63–92, 2008.

[5] F. Buttle. *Customer Relationship Management*. Butterworth-Heinemann, 1st edition, 2003.

[6] H. Cheng and E. Cantú-Paz. Personalized click prediction in sponsored search. In *WSDM*, pages 351–360, 2010.

[7] A. Clauset, M. E. J. Newman, and C. Moore. C moore: Finding community structure in very large networks. *http://www.arxiv.org/abs/condmat/0408187*, 2004.

[8] I. Cunha, R. Teixeira, D. Veitch, and C. Diot. Predicting and tracking internet path changes. In *SIGCOMM*, pages 122–133, 2011.

[9] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 271–280, New York, NY, USA, 2007. ACM.

[10] K. Dasgupta, R. Singh, B. Viswanathan, D. Chakraborty, S. Mukherjea, A. A. Nanavati, and A. Joshi. Social ties and their relevance to churn in mobile telecom networks. In *EDBT*, pages 668–677, 2008.

[11] P. Domingos. MetaCost: a general method for making classifiers cost-sensitive. In *KDD*, pages 155–164, 1999.

[12] P. Domingos. A few useful things to know about machine learning. *Commun. ACM*, 55(10):78–87, 2012.

[13] G. Dror, D. Pelleg, O. Rokhlenko, and I. Szpektor. Churn prediction in new users of yahoo! answers. In Mille et al. [26], pages 829–834.

[14] G. Dror, D. Pelleg, O. Rokhlenko, and I. Szpektor. Churn prediction in new users of yahoo! answers. In Mille et al. [26], pages 829–834.

[15] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *KDD*, pages 109–117, 2004.

[16] P. Greenberg. *CRM at the speed of light: Capturing and keeping customers in Internet real time*. Mcgraw-Hill, 4th edition, 2009.

[17] I. Guyon, V. Lemaire, M. Boullé, G. Dror, and D. Vogel. Analysis of the kdd cup 2009: Fast scoring on a large orange customer database. *Journal of Machine Learning Research - Proceedings Track*, 7:1–22, 2009.

[18] J. Hamilton. *Time series analysis*, volume 2. Cambridge Univ Press, 1994.

[19] S. Huang, M. Chen, B. Luo, and D. Lee. Predicting aggregate social activities using continuous-time stochastic process. In *CIKM*, pages 982–991, 2012.

[20] M. Jahrer, A. Toscher, J.-Y. Lee, J. Deng, H. Zhang, and J. Spoelstra. Ensemble of collaborative filtering and feature engineered models for click through rate prediction. In *KDDCup Workshop*, 2012.

[21] M. Karnstedt, M. Rowe, J. Chan, H. Alani, and C. Hayes. The effect of user features on churn in social networks. 2011.

[22] J. Kawale, A. Pal, and J. Srivastava. Churn prediction in mmorpgs: A social influence based approach. In *CSE (4)*, pages 423–428, 2009.

[23] K. Lerman, S. Intagorn, J.-H. Kang, and R. Ghosh. Using proximity to predict activity in social networks. In Mille et al. [26], pages 555–556.

[24] J. Lin and A. Kolcz. Large-scale machine learning at twitter. In *SIGMOD Conference*, pages 793–804, 2012.

[25] H. Ma, T. C. Zhou, M. R. Lyu, and I. King. Improving recommender systems by incorporating social contextual information. *ACM Trans. Inf. Syst.*, 29(2):9, 2011.

[26] A. Mille, F. L. Gandon, J. Misselis, M. Rabinovich, and S. Staab, editors. *Proceedings of the 21st World Wide Web Conference, WWW 2012, Lyon, France, April 16-20, 2012 (Companion Volume)*. ACM, 2012.

[27] T. P. Minka. A comparison of numerical optimizers for logistic regression. *Technical Report*, 2003.

[28] R. J. Oentaryo, E.-P. Lim, D. Lo, F. Zhu, and P. K. Prasetyo. Collective churn prediction in social network. In *ASONAM*, pages 210–214, 2012.

[29] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010.

[30] Y. Richter, E. Yom-Tov, and N. Slonim. Predicting customer churn in mobile networks through analysis of social groups. In *SDM*, pages 732–741, 2010.

[31] P. Sarkar, D. Chakrabarti, and M. I. Jordan. Nonparametric link prediction in dynamic networks. In *ICML*, 2012.

[32] A. Sittig and M. Zuckerberg. Managing information about relationships in a social network via a social timeline, 2010. US Patent 7,725,492.

[33] P. Smith and Z. Zook. *Marketing communications: integrating offline and online with social media*. Kogan Page, 2011.

[34] J. Subhlok, J. M. Stichnoth, D. R. O'hallaron, and T. Gross. Exploiting task and data parallelism on a multicomputer. *ACM SIGPLAN Notices*, 28(7):13–22, 1993.

[35] B. Viswanath, A. Mislove, M. Cha, and P. K. Gummadi. On the evolution of user interaction in facebook. In *WOSN*, pages 37–42, 2009.

[36] C. Wilson, B. Boe, A. Sala, K. P. N. Puttaswamy, and B. Y. Zhao. User interactions in social networks and their implications. In *EuroSys*, pages 205–218, 2009.

[37] J. Yang, X. Wei, M. S. Ackerman, and L. A. Adamic. Activity lifespan: An analysis of user survival patterns in online knowledge sharing communities. In *ICWSM*, 2010.

[38] T. Yang, R. Jin, Y. Chi, and S. Zhu. Combining link and content for community detection: a discriminative approach. In *KDD*, pages 927–936, 2009.

[39] Y. Zhu, E. Zhong, Z. Lu, and Q. Yang. Feature engineering for place category classification. In *Nokia Mobile Data Challenge Workshop, in conjunction with Pervasive'12*, 2012.