IEICE TRANS. INF. & SYST., VOL.E95–D, NO.12 DECEMBER 2012

PAPER

Implicit Influencing Group Discovery from Mobile Applications Usage*

Masaji KATAGIRI[†,††a)] ***and*** **Minoru ETOH**[†,†††], ***Members***

SUMMARY This paper presents an algorithmic approach to acquiring the influencing relationships among users by discovering implicit influencing group structure from smartphone usage. The method assumes that a time series of users' application downloads and activations can be represented by individual inter-personal influence factors. To achieve better predictive performance and also to avoid over-fitting, a latent feature model is employed. The method tries to extract the latent structures by monitoring cross validating predictive performances on approximated influence matrices with reduced ranks, which are generated based on an initial influence matrix obtained from a training set. The method adopts Nonnegative Matrix Factorization (NMF) to reduce the influence matrix dimension and thus to extract the latent features. To validate and demonstrate its ability, about 160 university students voluntarily participated in a mobile application usage monitoring experiment. An empirical study on real collected data reveals that the influencing structure consisted of six influencing groups with two types of mutual influence, i.e. intra-group influence and inter-group influence. The results also highlight the importance of sparseness control on NMF for discovering latent influencing groups. The obtained influencing structure provides better predictive performance than state-of-the-art collaborative filtering methods as well as conventional methods such as user-based collaborative filtering techniques and simple popularity.

key words: *influence, latent structure, user behavior modeling, mobile application, android, viral marketing*

1. Introduction

A recent study shows that asymmetric inter-personal influence may play an important role in understanding user adoption patterns and creating item recommendations with significantly better performance [2]. Given that user B tends to purchase products that user A has already bought, we should extract the *implicit* influencing relationship $A \rightarrow B$. Implicit influence includes indirect influence (influence through mediators with no direct interaction between A and B) as discussed in Liu's work [3] and also tendency effects such as user A simply tending to purchase at an earlier stage since he/she prefers innovative products, as well as direct influence through interaction with surrounding friends like A and B. This paper shows how to obtain inter-personal implicit influence factors by observing only purchase histories.

Related works on influence factors include several models that simulate information diffusion through a network. Widely-used models include the independent cascade (IC) model and the linear threshold (LT) model; both are fundamental probabilistic models of information diffusion [4], [5]. The IC model assumes that the social network structure is given and requires diffusion probabilities associated with links to be specified as input parameters. The LT model also assumes that the social network structure is given. In the LT model, each edge is assigned a weight. When the sum of weights exceeds a probabilistic threshold variable, diffusion occurs on edges from surrounding nodes that have been activated by information. Based on these models, several methods have been proposed; they estimate diffusion probabilities as a maximization problem [6]–[9]. Besides the IC/LT model, Song proposed the prediction of future adoption by the information flow network [10], which can be created from adoption history. Kawamae proposed personal innovator degree [2], which assumes that influence decreases exponentially as time passes, to predict future adoption. All of those published influence-related methods need to estimate each inter-personal influence factor independently. The number of factors to be estimated equals the number of edges of the given social network, since diffusion is assumed to occur only through edges of the social graph. For *implicit* influencing relationships, however, the diffusion network cannot be obtained beforehand, since it includes indirect influences and tendency effects. Thus, we cannot apply those methods if the number of users is high, due to the high computational complexity required to estimate each influence factor independently for every pair of users. Moreover, there exists another difficulty with the influence modeling approach. Since only common items adopted by both users can be used to acquire influence factors, the number of adopted items available for calculation tends to be small. This causes rough and unstable estimation for each influence probability.

In order to cope these difficulties, the authors propose to apply the latent feature model through matrix factorization. The recent success of collaborative filtering based on matrix factorization motivated the authors into applying the technique for influence modeling. Considering influence factors as probabilities on the context of information diffusion, the factor should be non-negative. Thus the authors employ NMF (non-negative matrix factorization) [11], [12] as a promising matrix factorization method.

Manuscript received January 27, 2012.
Manuscript revised July 10, 2012.

[†]The authors are with R&D Center, NTT DOCOMO, Inc., Yokosuka-shi, 239–8536 Japan.
[††]The author is with the Graduate School of Information Science and Technologies, Osaka University, Suita-shi, 565–0871 Japan.
[†††]The author is with Cybermedia Center, Osaka University, Ibaraki-shi, 567–0047 Japan.

*This paper is an extended and revised version of the authors' published conference paper [1].

a) E-mail: katagirim@nttdocomo.co.jp

DOI: 10.1587/transinf.E95.D.3026

NMF has emerged as a powerful tool since it offers data analysis with enhanced interpretability. It was originally proposed as a method for identifying matrix factors with part-of-whole interpretations [11]. NMF was later successfully applied to feature extraction in various fields such as music analysis, text mining, and gene analysis as well as image processing, see Weixiang's survey [12]. Moreover, a recent study proposed a community discovery method based on NMF [13].

We validate the model through an extensive empirical study at Osaka University involving 160 students. The study reveals several important findings as follows;

- Sparseness control by regularization is important — sparse feature vectors achieve better predictive performance
- Sparse feature vectors imply latent group structure in influencing relationship

Figure 1 shows the influencing structure acquired from the actual smartphone usage data. Each node and its color represents a user and his/her influencing group (G1 through G6). Each edge represents a major influencing relationship whose strength exceeds a threshold. Here, influencing groups are determined by latent features on influencing pattern rather than connectivity and/or density of edges. Therefore, it is different from community discovery on influencing relationship. Since intra-group influences tend to be strong compared with inter-group influences, the figure looks like a result of community discovery. However, it should be noted that the basic meaning is totally different. Further details of Fig. 1 are described in Sect. 3.3.

The authors have selected application usage of smartphones as the target of this implicit influence analysis. The reason is the increasing importance of understanding users' behavior on smartphones, since they are being rapidly adopted and appear likely to replace traditional cellular phones globally. The method described in this paper can be applied to any adoption history, not just application usage of smartphones.

To the best of our knowledge, this is the first paper to describe an attempt to apply the latent feature model to handling user influence relationships with validation by actual user data. Noteworthy contributions of this paper are as follows;

- Proposing a latent feature model on inter-personal implicit influence relationships that offers better predictive performance
- An extensive empirical study on real smartphone usage data involving 160 Osaka University students over a 4 month period; the results reveal latent group structure, confirm the importance of sparseness control over influence matrices, and demonstrate the proposals superiority over existing state-of-the-art methods

The authors verify its validity and effectiveness through an empirical study on actual, but specific, data. Although it is obvious that the model's validity and effectiveness depends on data, i.e. user behaviors, the authors believe this work does provide insights on a model that is worthy of being tested for similar setting tasks.

The structure of the rest of the paper is as follows. Section 2 overviews the problem and the model formulation. Section 3 explains the data collected and the experiments conducted. The experiments include latent group discovery and predictive performance evaluation. Section 4 provides additional discussions. Section 5 concludes the paper.

2. Inter-Personal Influence

2.1 Asymmetric Relationship

Figure 2 shows a simple example of the application adoption logs of 3 users (user 1, 2, 3) with 5 applications. Here, the horizontal axis represents time. A circle with number represents that the user adopted the corresponding application at that time. It is obvious the user-application relationships yields cascades, sequences of users who adopted an application (blue solid arrows). The goal is to predict future application adoption on each user for personalized recommendations. In the typical collaborative filtering (CF) approach, the next adoption by user 1 is predicted from the data of user 2 and user 3 since they have same degree of similarity as user 1 and, of key importance, CF does not take the order into account. Thus, applications 4 and 5 will be recommended equally by CFs. However, if we attend to the order of adoption, it is obvious that user 2 precedes user 1, while user 3 follows user 1. Thus user 1 is more likely to adopt application 5 than application 4. This clearly illustrates the importance of incorporating inter-personal influence when

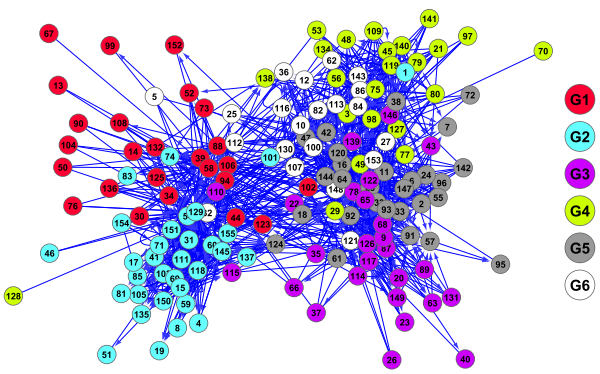


Fig. 1 Influencing relationship and discovered latent groups.

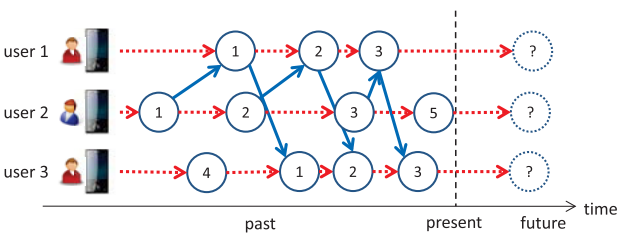


Fig. 2 Cascades in application usage.

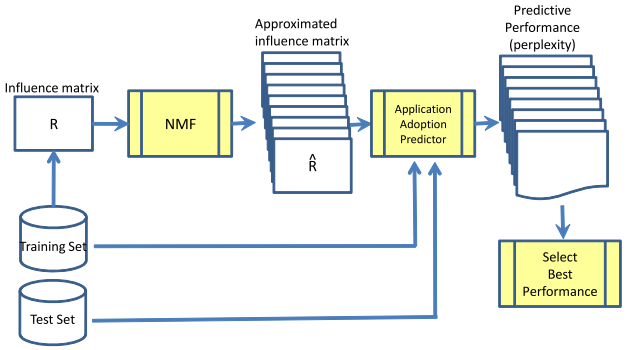


Fig. 3 Process overview of latent group model acquisition.

making predictions.

2.2 Overview of Discovering Influencing Groups

Figure 3 overview the influencing group discovery process targeted by this paper. Based on a training set, initial influence matrix $\mathbf{R}$ is calculated. Next, $\mathbf{R}$ is fed to matrix factorization (NMF) to generate approximated influence matrices $\widehat{\mathbf{R}}$ with reduced ranks. The Application Adoption Predictor predicts future application adoption for each user using $\widehat{\mathbf{R}}$ and the training set. Predictive performance is calculated by comparing predictive results with the test set data in the cross-validation manner. The following sections will detail each component.

2.3 Influence Factors from Observations

In this section, the authors define the influence factors for $\mathbf{R}$, which is similar to the static model described in Goyal's work [9]. The key difference is that we do not assume the existence of a given social network, since we cannot observe it in reality. Therefore, we need to predict influence factors for all users from observed application usage data.

2.3.1 Frequency-Based Influence Modeling

Under this model, user u has static influence probability $P_r(u \rightarrow v)$ for user v. After user u invokes application d, he/she tries to influence his/her inactive (yet to invoke d) neighbor v anytime within a certain time period τ. Here, parameter τ is assumed to be constant. Each attempt is viewed as a Bernoulli trial, so the Maximum Likelihood Estimator of successful probability is the ratio of number of successful attempts over the total number of trials. Hence, the influence probability of u on v is estimated as;

$$P_r(u \rightarrow v) = \frac{||A_{u \rightarrow v}||}{||A_u||}, \tag{1}$$

where $||A_u||$ denotes the number of applications which user u invoked in the training set. $||A_{u \rightarrow v}||$ denotes the number of applications which user u invoked then v invoked in the training set.

Let $P_{inf}(d|u)$ be a joint influence probability for application d on user u from surrounding neighbors. In this paper, we assume that the probability of various friends influencing u is independent of each other. Hence, the joint probability $P_{inf}(d|u)$ can be defined as follows;

$$P_{inf}(d|u) = \left[1 - \prod_{u'} \{1 - P_r(u' \rightarrow u) Y(u', d)\}\right], \text{ and} \tag{2}$$

$$Y(u', d) = \begin{cases} 1 & \text{if user } u' \text{ has invocated application } d \\ & \text{within } \tau \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

2.3.2 Influence Modeling Improvement by Entropy

There is a wide variety of applications, i.e., from very popular items to niche items. The latter have inherently higher possibilities of spreading through personal influence. Taking this into account, the authors propose an entropy-based influence factor. Entropy can be considered as an indicator of rarity. In the modeling proposal, we obtain the application entropy from the number of its unique users. Hence, the proposed entropy-based influence probability $\widehat{P}_r(u \rightarrow v)$ is defined as follows;

$$\widehat{P}_r(u \rightarrow v) = \frac{\sum_{a \in A_{u \rightarrow v}} \left(-log\left(\frac{u_a}{U_{glob}}\right)\right)}{\sum_{a \in A_u} \left(-log\left(\frac{u_a}{U_{glob}}\right)\right)}, \tag{4}$$

where U_{glob} denotes the total possible number of users for applications (Android applications in the experiment) and u_a denotes the number of unique users of application a. It is difficult to know the true values of u_a and U_{glob}. As a substitute for the true value of u_a, the authors apply the approximate number of downloads recorded by "Android Market". For U_{glob}, the tentative value of 2,000,000 was applied in the experiment. In order to obtain $P_{inf}(d|u)$ on $\widehat{P}_r(u' \rightarrow u)$, Eq. (2) has no need to be modified if $\widehat{P}_r(u' \rightarrow u)$ is applied instead of $P_r(u' \rightarrow u)$.

The authors performed pre-evaluation experiments that compared frequency-based modeling and entropy-based modeling. The results showed that the latter has better performance with slight margins. Therefore, entropy-based modeling is used in the later part of this paper. The effects of parameter τ were also been pre-evaluated. The results showed that longer is the better under the limited conditions examined. Therefore, all experiments in this paper were carried out under the condition of $\tau = \infty$.

Let user influence matrix $\mathbf{R}$ be a $U_{exp} \times U_{exp}$ matrix which has value of $P_r(u \rightarrow v)$ as an element for row u and column v, where U_{exp} denotes the number of subjects participating in the experiment.

2.4 Latent Feature Model

Recently proposed state-of-the-art CFs employ the latent

Table 1 NMF algorithms.

#	Ref.	Description
1	[17]	Standard NMF, based on Kullbach-Leibler divergence
2	[18]	Standard NMF, based on Euclidean distance
3	[19]	Non-smooth NMF based on Kullbach-Leibler divergence
4	[20]	Modified version of [18] based on Euclidean distance
5	[21]	Pattern-Expression NMF based on Euclidean distance
6	[22]	Alternating Least Square (ALS) approach

feature model based on matrix factorization [14]. In case of CF, the user–item matrix is factorized, so that it acquires latent user feature and latent item feature. Here, the authors propose to factorize the user influence matrix $\mathbf{R}$ (user–user matrix) by NMF.

NMF decomposes one non-negative matrix $\mathbf{X}$ ($n \times m$ matrix) into two non-negative matrices $\mathbf{A}$ ($n \times k$ matrix) and $\mathbf{B}$ ($k \times m$ matrix) with reduced dimension k ($k \ll n, m$) as a result of factorization, such that $\mathbf{X} \simeq \mathbf{AB}$. Here, let us call matrix $\mathbf{AB}$ ($n \times m$ matrix) the approximated matrix of $\mathbf{X}$ with reduced rank k; we depict it as $\widehat{\mathbf{X}}^{(k)}$. NMF has the unique feature of keeping values non-negative through decomposition as a constraint.

Subjecting user influence matrix $\mathbf{R}$ to NMF yields approximated matrix $\widehat{\mathbf{R}}^{(k)}$ with reduced rank k. Since $\mathbf{R}$ and $\widehat{\mathbf{R}}^{(k)}$ are matrices of the same size, the method to predict users' application adoptions by the influence matrix $\mathbf{R}$ can be applied to $\widehat{\mathbf{R}}^{(k)}$ without any change.

To find optimal operational boundary in terms of prediction performance, we tested several NMF variations with parameter search; NMF requires an inherently non-linear optimization process and so outputs different decomposition results. Table 1 shows the list of NMF algorithms used for the experiments. All NMF computations were performed using the package NMF [15] in R [16]. By observing the optimal operational boundary in prediction performance by approximated influence matrices $\widehat{\mathbf{R}}^{(k)}$ while sweeping rank k, we are able to find the most likely number of the latent feature dimension of the inter-personal relationship.

3. Data and Experiments

3.1 Collected Data

The authors have been monitoring the usage of smartphones in collaboration with Osaka University. 160 university students voluntarily participated in the monitoring process. Possible participants were requested to apply in a group of less than 50 students. Consequently 160 students consisted of six communites (groups of firends). Experimental usage monitoring software was installed on each of the smartphones, Xperia®, provided to the students with informed consent on data collection and its use for research purposes.

The software captures and records each application invocation. The log data are anonymized and collected on a server through 3G networks. Each log record contains timestamp, anonymized user-id, and the package name of the application invoked.

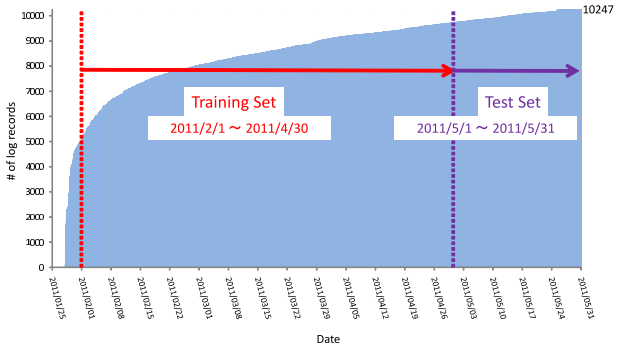


Fig. 4 Cumulative number of collected log records.

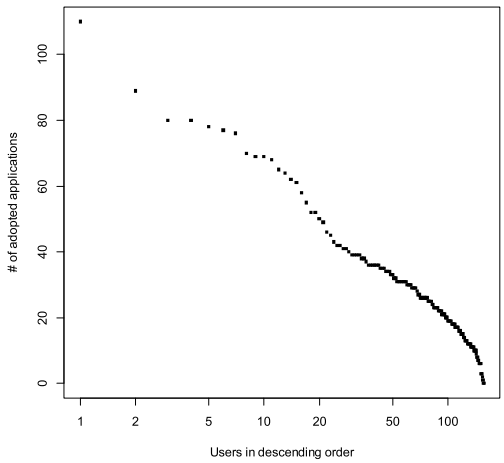


Fig. 5 Number of adopted applications by each user in descending order.

The first invocation records are extracted for each application and each user. Figure 4 shows the cumulative number of extracted records from beginning of the experiment. The authors split the data set into a training set and a test set by timestamp. The training set consists of records gathered over 89 days (February – April 2011) and the test set consists of records collected over 31 days (May 2011) right after the training set. The authors excluded records of applications that had less than 3 unique users in the training set. This yielded 3383 records (155 users and 291 applications) in the training set and 249 records (98 users and 116 applications) in the test set. All of the adopted applications were free.

Figure 5 shows the number of applications adopted by users in the training set in descending order. Maximum number of applications adopted was 110. Table 2 shows the demographic distribution of contributing students and the statistics of adopted applications. Figure 6 shows the number of shared adopted applications for all user pairs in descending order.

3.2 Performance Measure and Evaluation Process

Influence matrix $\mathbf{R}$ is estimated from the training set using Eq. (4). Approximated influence matrices $\widehat{\mathbf{R}}$ are calculated

Table 2 Demographic distribution of participating students.

Attributes		# of Students	Avg. Adopted APs
School	Letters	5	21.6
	Human Science	14	26.7
	Law	12	21.0
	Economics	13	38.5
	Science	7	24.9
	Medicine	23	25.7
	Pharma. Science	3	41.7
	Engineering	25	37.6
	Eng. Science	33	30.3
	Foreign Studies	19	22.4
	Others	1	13
Entrance Year	2007	3	29.0
	2008	50	26.2
	2009	28	34.5
	2010	74	28.9
Sex	F	46	24.4
	M	109	31.0
Total		155	29.1

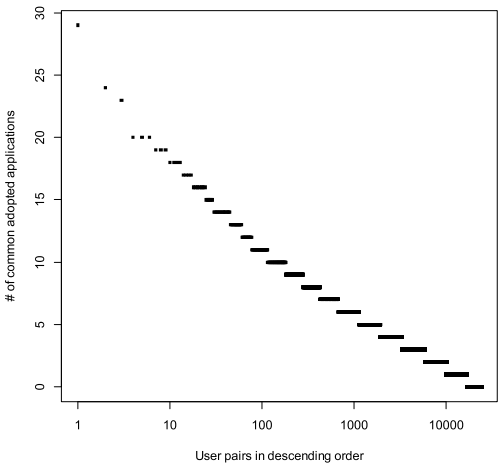


Fig. 6 Number of shared applications adopted by each user pair in descending order.

based on $\mathbf{R}$. Then application invocations during the test period are predicted using $\mathbf{R}$ or $\widehat{\mathbf{R}}$ with the usage history (the training set itself) and Eq. (2). The predictive performance is measured by comparing the predicted results with the observed test set data. Here, the authors employ (1) test set perplexity [23] and (2) mean average precision (MAP) [24], as performance measures. Test set perplexity is a standard measure for language model evaluation and corresponds to MAE/RMSE in the probabilistic domain. Since it demonstrates how well the model explains observations in terms of probability, it is applied for determining latent group structure. On the other hand, as a common performance measure for recommendation and information retrieval, MAP is employed for predictive performance comparison. Predictive performance is compared with state-of-the-art collaborative filtering methods such as biased-SVD [25], SVD++ [25], wALS [24] and NMF-based CF, as well as conventional user-based CF and simple popularity-based methods. The results will be shown in Sect. 3.5.

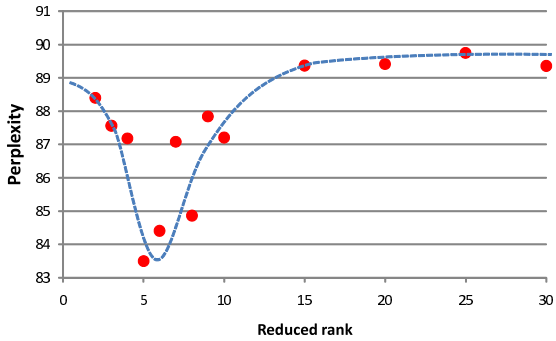


Fig. 7 Perplexity of influence matrices with reduced ranks created by NMF.

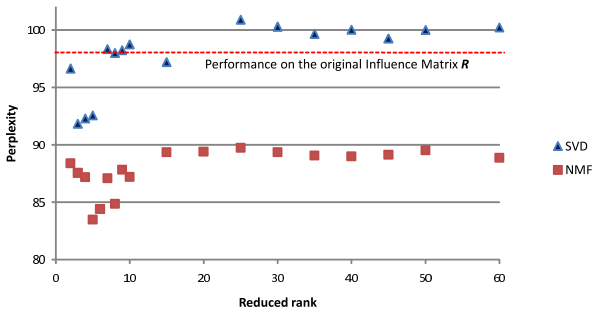


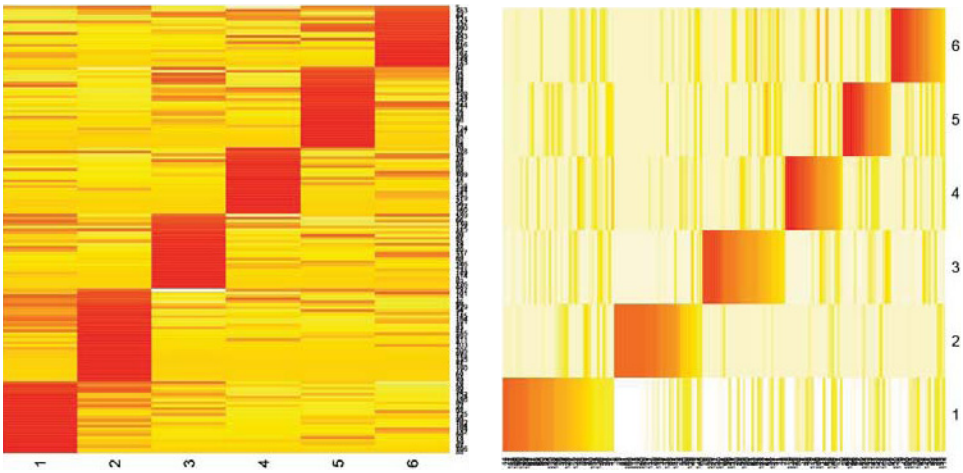
Fig. 8 Perplexity of influence matrices with reduced ranks created by SVD and NMF.

3.3 Influencing Groups Discovered by Matrix Factorization

Figure 7 plots the operational optimal boundary in predictive performances on approximated influence matrices $\widehat{\mathbf{R}}$ with reduced ranks by NMF. We can observe certain decrease of perplexity (i.e., performance improvement) around remarkably small rank of 5 or 6. This implies that the inter-personal relationship has 5 or 6 latent components (or groups) structure and each user's characteristic can be represented by weighting coefficients for these components.

Figure 8 plots the predictive performance results from the approximated influence matrix by SVD/NMF derived from the same data set used for Fig. 7. In any reduced rank, NMF demonstrates better perplexity than SVD. In the case of SVD, we observe worse perplexity than the original influence matrix used (which is 98.2635) over a wide range of reduced rank. It is notable that $\widehat{\mathbf{R}}$ by NMF retains its superior performance to the original $\mathbf{R}$, although there is no guarantee that the approximation leads to a predictive performance improvement, supposedly as a result of the non-negativity constraint. Moreover, Fig. 8 also shows that smaller reduced ranks produced lower perplexity values on $\widehat{\mathbf{R}}$ by SVD as well as by NMF. This gives us another piece of evidence that supports the latent group structure in the inter-personal relationship.

Figure 9 visualizes the decomposed influence matrices at the lowest perplexity (reduced rank $k = 6$). The values



(a) Coefficient matrix (b) Basis matrix

Fig. 9 Heatmaps of decomposed influence matrices ($k = 6$).

Table 3 Inter-group / Intra-group influence.

# of persons (proportion in %)		Influencing group where being influenced persons belong to					
		G1	G2	G3	G4	G5	G6
Influencing group where influencing persons belong to	G1	20 (51.3%)	5 (12.8%)	3 (7.7%)	6 (15.4%)	2 (5.1%)	3 (7.7%)
	G2	1 (3.2%)	24 (77.4%)	1 (3.2%)	2 (6.5%)	2 (6.5%)	1 (3.2%)
	G3	2 (6.9%)	0 (0.0%)	11 (37.9%)	4 (13.8%)	10 (34.5%)	2 (6.9%)
	G4	0 (0.0%)	2 (10.0%)	3 (15.0%)	7 (35.0%)	2 (10.0%)	6 (30.0%)
	G5	1 (5.9%)	0 (0.0%)	5 (29.4%)	4 (23.5%)	6 (35.3%)	1 (5.8%)
	G6	1 (5.3%)	1 (5.3%)	3 (15.8%)	0 (0.0%)	6 (31.6%)	8 (42.1%)

of matrix elements are normalized by each user (the sum of elements is 1), and are expressed by color (red means high value, light yellow small value). Figure 9 (a) shows the coefficient matrix on influencing patterns and Fig. 9 (b) shows basis matrix representing the patterns of being influenced. Vertical (horizontal) axis represents individual users in the coefficient (basis) matrix. Vector of each user on the coefficient matrix denotes his/her influencing pattern, it indicates who are influenced by him/her. On the other hand, vector of each pattern (1 to 6 in this case) on the basis matrix shows the destination of influence. For example, pattern 1 means heavy influence is given to the 39 students at the left most columns of the basis matrix. Please note that those matrices are directional, i.e., influencing direction and being influenced direction. The matrix of Fig. 9 (b) can be considered as coefficient matrix of being influenced patterns, in such a case, Fig. 9 (a) is considered as a basis matrix as well.

Here, we can see each user has one dominant component according to Fig. 9 (a). The users who share the same dominant component in Fig. 9 (a) have similar influencing patterns. Let us call a group of users whose members share the dominant component (i.e. influencing pattern) an "influencing group". On the same way, let us call a group of users whose members share dominant component in Fig. 9 (b) (i.e. being influenced pattern) an "influencee group". A user belonging to influencing group G1 gives influence mainly to members of influencee group g1. Moreover, each user belongs to both of an influencing group and an influencee

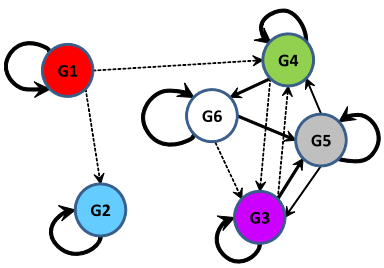


Fig. 10 Visualized relationship among influencing groups.

group. In other words, each member of influencee group g1 also belongs to one of influencing groups. Thus, we can calculate how strongly the members of influencing group G1 influence each influencing group. Based on the above consideration, the member proportions are shown in Table 3. That is, the table represents inter-group/intra-group influence proportion. According to Table 3, influence relationship among influencing groups can be roughly visualized as shown by Fig. 10. Table 3 shows that intra-group influence tends to occupy the major part. That means mutual influence among members within same influencing group is relatively dense. In addition, some of the inter-group influence relationships are also dense. Moreover, from Fig. 10, we can identify rough inter-group structures, i.e. G3–G6 are tightly linked while G1 and G2 are more independent from G3–G6.

Figure 1 is the result of applying graph structure visualization to $\widehat{\mathbf{R}}$. Top 6% of influencing user–user relations were

Table 4 NMF algorithms contributing to Fig. 7 plots.

Reduced Rank	Best Algorithm
2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20	Non-smooth NMF based on Kullbach-Leibler divergence (nsNMF)
25, 30, 35, 40, 45, 50, 60	Pattern-Expression NMF based on Euclidean distance (PE-NMF)

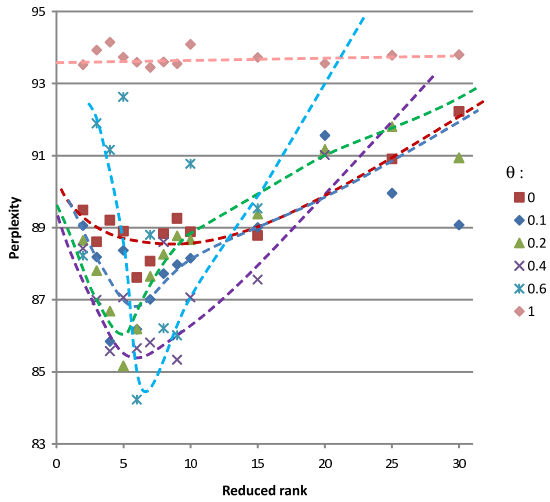


Fig. 11 Perplexity vs parameter θ of nsNMF.

extracted from $\widehat{\mathbf{R}}$, where the sum of the extracted influencing factors occupies 35% of the grand total. A commonly used force-directed layout algorithm produced Fig. 1 from the extracted user–user relations with each node colored by identified influencing group mentioned above. It is obvious that Fig. 10 is fairly consistent in explaining the layout of Fig. 1.

3.4 Effects of Sparseness Control

Table 4 presents NMF algorithms which produced the best predictive performances plotted in Fig. 7. As we can see, the algorithm "nsNMF" performs the best in terms of perplexity for all dimensions lower than 20. "nsNMF" is an algorithm that focuses on controlling the sparseness on decomposed matrices while retaining its good approximation of the original matrix as well. This result shows that sparseness control is one of key factors in discovering the latent group structure of implicit influence.

"nsNMF" has one control parameter θ ($0 \leq \theta \leq 1$). $\theta = 1$ is the tightest sparseness constraint. With $\theta = 0$, the algorithm performs as ordinary NMF. Figure 11 shows predictive performances (perplexity) for different θ values. The curve is broad when $\theta = 0$ but becomes steep as θ increases. When θ exceeds 0.6, no drop in perplexity observed. Making decomposed matrices sparse means to narrow the ranges of people who influence / are influenced. This implies a mechanism similar to that of a resonant circuit. θ corresponds to the Q value of the circuit. With increased Q value (θ) below an upper limit, the target characteristic becomes sharper. Since the dimension / rank has discrete value, it is reasonable that there be an upper bound in terms of Q value (θ) where resonance can be observed.

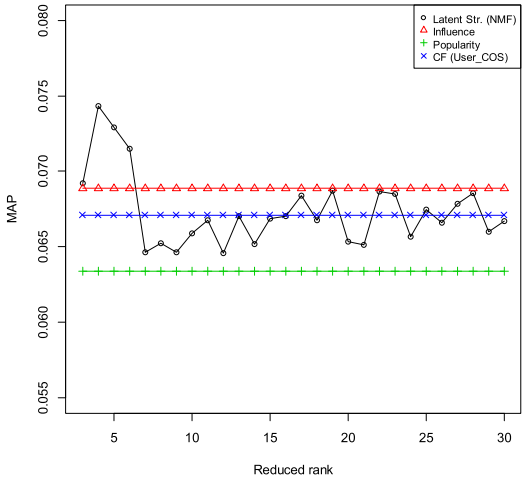


Fig. 12 Predictive performance (MAP) vs. reduced rank.

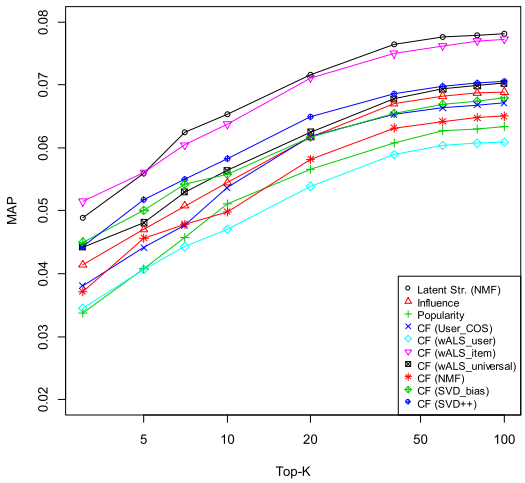


Fig. 13 Predictive performance comparison (MAP).

3.5 Predictive Performance Comparison

Figure 12 shows predictive performances of the proposed latent feature model with varied reduced rank. As reference, performances of several popular recommendation techniques are also shown. In Fig. 12, "Latent Str. (NMF)" denotes the latent feature model discussed in the previous section. "Influence" denotes prediction based on $\mathbf{R}$ without low rank approximation. "CF (User_COS)" denotes the conventional user-based CF based on cosine similarity. "Popularity" denotes simple prediction based on popularity, which is measured by the numbers of unique users in the training period. Similar to Fig. 7, we can observe a performance increase around *rank* = 6 for the latent feature model.

Figure 13 shows a performance comparison with various CF schemes. "CF (NMF)" uses a latent feature model which employs NMF for matrix factorization. "CF

(SVD_bias)" and "CF (SVD++)" are SVD-based CFs [25] and are known to be high performers for movie rating prediction. "CF (wALS_item)","CF (wALS_user)" and "CF (wALS_universal)" are specialized CFs for the one-class setting [24]. Here, it should be noted that the task under discussion is one-class setting, where no negative samples are available for training. Thus "CF (NMF)", "CF (SVD_bias)" and "CF (SVD++)" were trained and tested with AMAN (all missing as negative) strategy.

As we can see, the proposed latent feature model demonstrates the best performance, slightly higher than "wALS_item". This result validates the proposed model and its usefulness for personalized recommendations as one of its applications.

4. Discussion

4.1 Latent Group Characteristics

The latent groups (influencing groups and influencee groups) are formed by similar influencing / being influenced patterns. This implies that the group members tend to have similar application usage behavior and demographic attributes. This motivated us to perform an analysis of demographics / application usage and influencing groups.

Some demographic analysis results are shown in Table 6. Some degree of correlation was observed for schools and sex. The top 20 most-frequently-used applications in each influencing group are listed in Table 7. Since the number of groups was selected by best predictive performance and not for better topic separation, it is hard to find concrete topics or characteristics on application preferences for each influencing group. Some symbolic applications can be found such as medicine search application in G4 to which many students of the medicine school belong.

Table 5 Correlation ratios.

Factor	Correration ratio	
	$\mathbf{R}$	$\widehat{\mathbf{R}}$
User pair in Same School vs others	0.020142	0.017418
User pair in Same Grade vs others	0.006050	0.001535
User pair in Same Community vs others	0.019159	0.018326
Pair of Heavy Users vs others	0.068715	0.098507
Pair of Light Users vs others	0.063818	0.078679
User pair in Same Influencing Gr. vs others	0.092977	0.208375

4.2 Correlation Analysis with Other Factors

If some user attributes exhibit strong correlation with implicit influence, they could contribute to the improvement in predictive performance. Thus, the authors conducted a correlation analysis on influence factors with some user attributes. All user pairs were classified into two groups depending on whether both users have same attribute or not. Then correlation ratios were calculated between the groups. Tested attributes were school, entrance year, community, and whether the number of adopted applications is higher/lower than average. Here, community means a group of friends, as mentioned in Sect. 3.1

The results are shown in Table 5. No obvious correlation was observed on any attributes except acquired influencing groups. This implies that it is difficult to infer implicit influencing relationship by such attributes.

4.3 Visualizing Approximation Magnitude

From viral marketing point of view, it is valuable to identify persons who are opinion leaders that influence their surroundings, in order to make targeting strategies more effective. Here, each user's degree of influence, $influentiality$, and susceptibility to influence, $influenceability$, can be defined as follows;

$$influentiality_u = \frac{\Sigma_v P_r(u \to v)}{U_{exp} - 1}, \quad \text{and} \tag{5}$$

$$influenceability_u = \frac{\Sigma_v (P_r(v \to u))}{U_{exp} - 1}. \tag{6}$$

Figure 14 shows a scatter plot of $influentiality$ versus $influenceability$ calculated from the original influence matrix $\mathbf{R}$ and the approximated influence matrix $\widehat{\mathbf{R}}$ with the best predictive performance. Each plot corresponds to an individual user. Arrows show how much the approximation changes user position. We can see that there is no obvious correlation in distribution between $influentiality$ and $influenceability$. We can also see several outliers. In the case of users who used few applications, the values of $P_r(u \to v)$ becomes unstable. This intuitively explains the

Table 6 Demographic distribution of influencing groups.

Influencing Group	School											Sex		Total
	Letters	Human Science	Law	Economics	Science	Medicine	Pharmaceutical Sciences	Engineering	Engineering Science	Foreign Studies	Others	F	M	
G1	1	3	2	2	1	5	1	4	3	3	0	11	14	25
G2	1	6	3	2	1	3	1	3	6	6	0	10	22	32
G3	2	1	2	0	3	1	1	4	10	1	1	2	24	26
G4	0	0	2	3	0	8	0	1	6	3	0	10	13	23
G5	1	2	1	4	2	3	0	8	4	3	0	5	23	28
G6	0	2	2	2	0	3	0	5	4	3	0	8	13	21
Total	5	14	12	13	7	23	3	25	33	19	1	46	109	155

Table 7 Popular applications in influencing groups.

Rank	Influencing Group					
	G1	G2	G3	G4	G5	G6
1	com.android.calculator2	Android	com.android.inputmethod.latin	com.cookpad.android.activities	com.cooliris.media	com.google.android.voicesearch
2	com.android.inputmethod.latin	com.google.android.providers.enhancedgoogle search	net.binzume.android.nicoplayer	com.moxier.mail	com.android.packageinstaller	com.sonyericsson.android.basicwords
3	com.facebook.katana	com.sonyericsson.android.timescape	com.android.alarmclock	cn.bluesky.neatreversi	com.android.wallpaper.livepicker	com.mobisystems.office
4	com.mobisystems.office	com.sonyericsson.conversations	com.android.packageinstaller	jp.co.gnavi.activity	com.facebook.katana	com.sonyericsson.pccompanion
5	com.sonyericsson.quadrapop	com.android.browser	com.google.android.youtube	com.android.calculator2	com.google.android.voicesearch	com.sonyericsson.quadrapop
6	com.android.vending	com.google.android.youtube	com.pdanet	com.google.android.apps.maps	com.google.zxing.client.android	com.spritemobile.backup.semc
7	com.cooliris.media	com.android.phone	com.skype.raider	com.sonyericsson.android.servicemenu	com.joelapenna.foursquared	jp.picolyl.led_light
8	com.sonyericsson.android.basicwords	com.google.android.apps.maps	com.adobe.reader	com.sonyericsson.quadrapop	com.nttdocomo.android.compass	com.android.calculator2
9	com.sonyericsson.android.iwnnime	com.mobisystems.office	com.android.calculator2	de.joergjahnke.mario.android.free	com.sonyericsson.android.friendpivot	com.cooliris.media
10	com.sonyericsson.android.servicemenu	com.sonyericsson.android.camera	com.cooliris.media	jp.co.c_lis.ccl.medicinesearch.android	com.sonyericsson.android.wallpaperchooser	com.facebook.katana
11	com.sonyericsson.setupwizard	com.sonyericsson.android.mediascape	com.evernote	jp.fuukiemonster.webmemo	com.sonyericsson.pccompanion	com.google.android.street
12	com.android.globalsearch	com.sonyericsson.android.socialphonebook	com.justsystems.atokmobile.trial.service	nikeno.Tenki	jqsoft.apps.mysettings	com.google.android.talk
13	com.android.wallpaper.livepicker	com.android.launcher	com.nttdocomo.android.compass	wni.WeathernewsTouch.jp	com.adobe.reader	com.nttdocomo.android.compass
14	com.nttdocomo.android.docomo_market	com.google.android.gm	com.sonyericsson.textinput.uxp	com.adobe.reader	com.android.alarmclock	com.sakura.News2010
15	com.sonyericsson.android.socialservicesetting	com.sonyericsson.search	jp.bustercurry.virtualtenho_g	com.android.alarmclock	com.android.htmlviewer	com.sonyericsson.android.servicemenu
16	com.sonyericsson.pccompanion	com.android.settings	jp.radiko.gui.main	com.android.globalsearch	com.android.music	com.sonyericsson.trackid3.client
17	com.sonyericsson.textinput.uxp	com.android.vending	org.adw.launcher	com.android.providers.applications	com.bumptech.bumpga	com.taptu.wapedia.android
18	com.android.alarmclock	com.google.android.voicesearch	asia.sonix.sekaimeigenwidget	com.bumptech.bumpga	com.clapfootgames.tankhero	de.shandschuh.sparserss
19	com.android.calendar	com.sonyericsson.android.friendpivot	com.android.phone	com.ekitan.android	com.donapon.pisces.cashbook	jp.co.c2inc.medicallite.first
20	com.cootek.touchpal	jp.co.sonyericsson.android.playnow	com.android.providers.applications	com.google.android.location	com.fgol.sharkfree	jp.co.labelgate.moratouch

appearance of the outliers.

influentiality and *influenceability* above are based on implicit influences, thus they reflect tendency effects as mentioned in Sect. 1. Therefore, a user with high *influentiality* does not necessarily mean a real opinion leader. The authors leave this accuracy issue for future work. Identifying opinion leaders may require further analytical studies.

5. Conclusion

This paper described an algorithmic approach that can discover implicit influencing group structures of social influence among users in terms of smartphone application usage. The method utilizes nonnegative matrix factorization with a sparseness constraint. Latent user groups were experimentally identified by observing predictive performance against reduced dimensions of the influence matrix. Analysis found the influencing relationships, which consist of intra-group influence and inter-group influence. Moreover, the model successfully predicted students' application downloads from their past activities; a MAP evaluation showed that it attained better performance than state-of-the-art and conventional collaborative filtering methods, as well

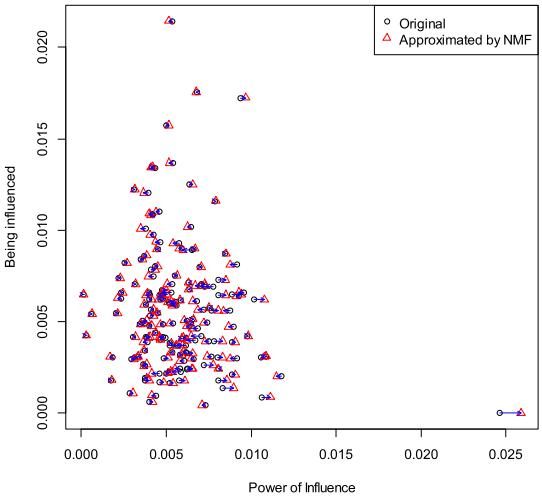


Fig. 14 Magnitude on influence matrix approximation.

as a popularity method.

Future studies should examine more sophisticated diffusion models to enhance performance. Scalability and adaptability are key goals to be tackled. Since influencing relationships depend on the topic, model extension for topic-based is also worth tackling.

Acknowledgments

The authors would like to thank Yuka Ikebe and Yoshikazu Akinaga for their assistance in collecting the data and visualizing the graph data, respectively.

References

[1] M. Katagiri and M. Etoh, "Social influence modeling on smartphone usage," Proc. 7th International Conference on Advanced Data Mining and Applications, ADMA 2011, Beijing, China, Part II, pp.292–303, Dec. 2011.

[2] N. Kawamae, H. Sakano, and T. Yamada, "Personalized recommendation based on the personal innovator degree," Proc. 3rd ACM Conference on Recommender Systems, RecSys '09, New York, USA, pp.329–332, Oct. 2009.

[3] L. Liu, J. Tang, J. Han, M. Jiang, and S. Yang, "Mining topic-level influence in heterogeneous networks," Proc. 19th ACM International Conference on Information and Knowledge management, CIKM '10, Tronto, Canada, pp.199–208, Oct. 2010.

[4] D. Kempe, J. Kleinberg, and E. Tardos, "Maximizing the spread of influence through a social network," Proc. 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '03, Washington D.C., USA, pp.137–146, Aug. 2003.

[5] J. Goldenberg, B. Libai, and E. Muller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," Marketing Letters, vol.12, no.3, pp.211–223, Aug. 2001.

[6] D. Gruhl, R. Guha, D. Liben-Nowell, and A. Tomkins, "Information diffusion through blogspace," Proc. 13th International Conference on World Wide Web, WWW '04, New York, USA, pp.491–501, May 2004.

[7] M. Kimura, K. Saito, and R. Nakano, "Extracting influential nodes for information diffusion on a social network," Proc. 22nd National Conference on Artificial Intelligence, AAAI-07, Vancouver, Canada, vol.2, pp.1371–1376, July 2007.

[8] K. Saito, R. Nakano, and M. Kimura, "Prediction of information diffusion probabilities for independent cascade model," in Knowledge-Based Intelligent Information and Engineering Systems, ed. I. Lovrek, R. Howlett, and L. Jain, Lect. Notes Comput. Sci., vol.5179, pp.67–75, Springer Berlin/Heidelberg, Sept. 2008.

[9] A. Goyal, F. Bonchi, and L.V. Lakshmanan, "Learning influence probabilities in social networks," Proc. 3rd ACM International Conference on Web Search and Data Mining, WSDM '10, New York, USA, pp.241–250, Feb. 2010.

[10] X. Song, B.L. Tseng, C.Y. Lin, and M.T. Sun, "Personalized recommendation driven by information flow," Proc. 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '06, Seattle, USA, pp.509–516, Aug. 2006.

[11] D.D. Lee and H.S. Seung, "Learning the parts of objects by non-negative matrix factorization," Nature, vol.401, no.6755, pp.788–791, Oct. 1999.

[12] L. Weixiang, Z. Nanning, and Y. Qubo, "Nonnegative matrix factorization and its applications in pattern recognition," Chinese Science Bulletin, vol.51, no.1, pp.7–18, Jan. 2006.

[13] F. Wang, T. Li, X. Wang, S. Zhu, and C. Ding, "Community discovery using nonnegative matrix factorization," Data Min. Knowl. Discov., vol.22, pp.493–521, May 2011.

[14] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," Computer, vol.42, no.9, pp.30–37, Sept. 2009.

[15] R. Gaujoux and C. Seoighe, "A flexible R package for nonnegative matrix factorization," BMC bioinformatics, vol.11, no.1, pp.367+, July 2010.

[16] R Development Core Team, R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2008.

[17] J.P. Brunet, P. Tamayo, T.R. Golub, and J.P. Mesirov, "Metagenes and molecular pattern discovery using matrix factorization," Proc. National Academy of Sciences, vol.101, no.12, pp.4164–4169, March 2004.

[18] D.D. Lee and H.S. Seung, "Algorithms for Non-negative Matrix Factorization," in Advances in Neural Information Processing Systems 13, ed. T.K. Leen, T.G. Dietterich, and V. Tresp, pp.556–562, The MIT Press, MA, USA, April 2001.

[19] A. Pascual-Montano, J. Carazo, K. Kochi, D. Lehmann, and R.D. Pascual-Marqui, "Nonsmooth nonnegative matrix factorization (nsnmf)," IEEE Trans. Pattern Anal. Mach. Intell., vol.28, no.3, pp.403–415, March 2006.

[20] L. Badea, "Extracting gene expression profiles common to colon and pancreatic adenocarcinoma using simultaneous nonnegative matrix factorization," Proc. Pacific Symposium on Biocomputing 2008, PSB 2008, Kohala Coast, USA, pp.267–278, Jan. 2008.

[21] J. Zhang, L. Wei, X. Feng, Z. Ma, and Y. Wang, "Pattern expression nonnegative matrix factorization: Algorithm and applications to blind source separation," Computational Intelligence and Neuroscience, vol.2008, April 2008.

[22] H. Kim and H. Park, "Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis," Bioinformatics, vol.23, no.12, pp.1495–1502, May 2007.

[23] S. Furui, "Speech and speaker recognition evaluation," in Evaluation of Text and Speech Systems, ed. L. Dybkjœr, H. Hemsen, W. Minker, and N. Ide, Text, Speech and Language Technology, vol.37, pp.1–27, Springer Netherlands, 2007.

[24] R. Pan, Y. Zhou, B. Cao, N.N. Liu, R.M. Lukose, M. Scholz, and Q. Yang, "One-class collaborative filtering," Proc. 8th IEEE International Conference on Data Mining, (ICDM 2008), Pisa, Italy, pp.502–511, Dec. 2008.

[25] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," Proc. 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD

’08, Las Vegas, USA, pp.426–434, Aug. 2008.

Masaji Katagiri is Director of Data Mining Group, Service & Solution Development Department at NTT DOCOMO, Japan, and also a Ph.D. student at Osaka University. He joined Nippon Telegraph and Telephone in 1986. Since 1999, he has been working for NTT DOCOMO. He was a visiting industrial fellow at the University of California at Berkeley from 1990 to 1991, and a lab director of DoCoMo USA Labs from 2002 to 2004. His current research interests include knowledge discovery, social network analysis, and mobile multimedia applications. He received his B. Engineering and M. Engineering degrees from Waseda University in 1984 and 1986, respectively. He is a member of IEEE, and IPSJ.

Minoru Etoh has the triple-roles as President & CEO of DOCOMO Innovations, Managing Director of Service & Solution Development Department (Vice President of NTT DOCOMO), and Visiting Professor at Osaka University. In 2006, he launched a big data mining project in DOCOMO. Since then, he has been leading that project for new service creation and enhancement, in addition to his business administration duties in California and Japan. His expertise covers a wide range of mobile multimedia, network architecture, terminal software, coding technologies, media transport, and data mining. He received his B.E. and M.S.E.E. from Hiroshima University, Ph.D. degree from Osaka University, in 1983, 1985 and 1993, respectively.