Published in "Europhysics Letters 80(6): 68003, 2007"
which should be cited to refer to this work.

Recommendation model based on opinion diffusion

Y.-C. Zhang[1,2], M. Medo[1(a)], J. Ren[1], T. Zhou[1,3], T. Li[2] and F. Yang[2]

[1] *Physics Department, University of Fribourg - 1700 Fribourg, Switzerland*
[2] *Department of Physics, Renmin University of China - Beijing 100872, PRC*
[3] *Department of Modern Physics, University of Science and Technology of China - Hefei 230026, PRC*

PACS 89.65.-s – Social and economic systems
PACS 89.75.Hc – Networks and genealogical trees
PACS 89.20.Ff – Computer science and technology

Abstract – Information overload in the modern society calls for highly efficient recommendation algorithms. In this letter we present a novel diffusion-based recommendation model, with users' ratings built into a transition matrix. To speed up computation we introduce a Green function method. The numerical tests on a benchmark database show that our prediction is superior to the standard recommendation methods.

Introduction. – The exponential growth of the Internet [1] and the World-Wide-Web [2] confronts us with the information overload: we face too many data and data sources, making us unable to find the relevant results. As a consequence we need automated ways to deal with the data. Recently, a lot of work has been done in this field. The two main directions of the research are correlation-based methods [3,4] and spectral methods [5]. A good overview of the achieved results can be found in [6,7].

Despite the amount of work done, the problem is not satisfactorily exploited yet as both the prediction accuracy and the computational complexity can be further improved. In this letter we propose a new method based on diffusion of the users' opinions in an object-to-object network. This method can be used for any data where users evaluate objects on an integer scale. Using data from a real recommender application (GroupLens project) we show that the present model performs better than the standard recommendation methods. In addition, a Green function method is proposed here to further reduce computation in some cases.

The model. – In the input data, we label the total number of users as M and the total number of objects as N (since we focus here on the movie recommendation, instead of the general term *object* we often use the term *movie*). To make a better distinction between these two groups, for user-related indices we use lower-case letters $i, j, k, \dots$ and for movie-related indices we use Greek letters $\alpha, \beta, \gamma, \dots$. We assume that users' assessments are given in the integer scale from 1 (very bad) to 5 (very good). The rating of user i for movie α we denote $v_{i\alpha}$. The number of movies rated by user i we label k_i. The rating data can be described by the weighted bipartite graph where the link between user i and movie α is formed when user i has already rated movie α and the link weight is $v_{i\alpha}$. Such a bipartite graph can give rise to two different types of graphs (often called *projections*): object-to-object and user-to-user. A general discussion on information networks can be found in [8], projections of bipartite graphs are closely investigated in [9,10].

The recommendation process starts with the preparation of a particular object-to-object projection of the input data. Projections usually lead to a loss of information. In order to eliminate this phenomenon, instead of merely creating a link between two movies, we link the ratings given to this pair of movies. As a result, we obtain 25 separate connections (channels) for each movie pair. This is illustrated in fig. 1 on an example of a user who has rated three movies; as a result, three links are created between the given movies. When we process data from all users, contributions from all users shall accumulate to obtain an aggregate representation of the input data: a weighted movie-to-movie network. From the methodological point of view, this model is similar to the well-known quantum diffusion process (see [11,12]).

To each user we need to assign a weight. In general, if user i has rated k_i movies, $k_i(k_i-1)/2$ links in the network are created (or fortified). If we set the user weight to $1/(k_i-1)$, the total contribution of user i is directly

(a) E-mail: matus.medo@unifr.ch

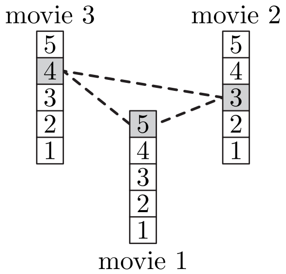

Fig. 1: Graphical representation of the links created by a user who has rated only movies 1 (rating 5), 2 (rating 3), and 3 (rating 4).

proportional to k_i, and this is a plausible premise[1]. Since the users who have seen only one movie add no links to the movie-to-movie network, the divergence of the weight $1/(k_i-1)$ at $k_i=1$ is not an obstacle.

Since between each pair of movies (α,β) we create multiple links, it is convenient to write their weights as a 5×5 matrix $\mathsf{W}_{\alpha\beta}$. Each rating can be represented by a column vector in 5-dimensional space: we represent rating $v_{i\alpha}=1$ as $\boldsymbol{v}_{i\alpha}=(1,0,0,0,0)^T$, rating $v_{i\alpha}=2$ as $\boldsymbol{v}_{i\alpha}=(0,1,0,0,0)^T$, and so forth. If the vote has not been given yet, we set $\boldsymbol{v}_{i\alpha}=(0,0,0,0,0)^T$. Then using the linking scheme from fig. 1 and the user weights $1/(k_i-1)$ we write

$$\mathsf{W}_{\alpha\beta}=\sum_{i=1}^{M}\frac{\boldsymbol{v}_{i\alpha}\boldsymbol{v}_{i\beta}^T}{k_i-1}, \tag{1}$$

where we sum contributions from all users. In this way we convert the original data represented by a weighted bipartite graph into a weighted object-to-object network.

The non-normalized weights $\mathsf{W}_{\alpha\beta}$ form a symmetric matrix W with dimensions $5N\times 5N$. By the column normalization of W we obtain an unsymmetric matrix Ω. It describes a diffusion process on the underlying network with the outgoing weights from any node in the graph normalized to unity (see also a similar diffusion-like process in [14] and the PageRank algorithm[2]).

Now we shall investigate the equation

$$\Omega\boldsymbol{h}=\boldsymbol{h}, \tag{2}$$

where $\boldsymbol{h}$ is a $5N$-dimensional vector (the first 5 elements correspond to movie 1, the next 5 elements to movie 2, etc.). Denote $n_{\alpha s}$ $(\alpha=1,\dots,M,\ s=1,\dots,5)$ the number of times movie α has been rated with the rating s. Here we exclude the votes given by the users who have rated only one movie because these users do not contribute to Ω. It is easy to prove that the vector

$$\boldsymbol{h}^*=(n_{11},\dots,n_{15},\dots,n_{N1},\dots,n_{N5})^T \tag{3}$$

is a solution of eq. (2). Moreover, the solution is unique up to multiplication by a constant and, as we will see later, all vectors in the form $\lambda\boldsymbol{h}$, $\lambda\neq 0$, lead to identical predictions. Denote $L:=\mathbb{1}-\Omega$ the Laplace matrix, the forementioned uniqueness of $\boldsymbol{h}^*$ is equivalent to $\operatorname{rank}(L)=5N-1$, which we prove in the following paragraph. It is worthwhile to emphasize that the unique solution $\boldsymbol{h}^*$ reproduces some features of the original input data, which strongly supports rationality and relevance of the construction of Ω.

Using elementary row/column operations one can shift all the rows/columns corresponding to the zero-rows/zero-columns of Ω to the bottom and right of L, leading to $\left(\begin{smallmatrix}L' & O\\ O & \mathbb{1}\end{smallmatrix}\right)$, where O and $\mathbb{1}$ are the zero and the identity matrix. The dimension of $\mathbb{1}$ we label as D, the dimension of L' is then $5N-D$. The matrix L' has four properties: i) All its diagonal elements are 1. ii) All its non-diagonal elements lie in the range $[-1,0]$. iii) The sum of each column is zero. iv) In each row, there is at least one non-diagonal non-zero element. One can prove that the rank of any matrix with these four properties is equal to its dimension minus one, $5N-D-1$ in this case. Since $\operatorname{rank}(\mathbb{1})=D$, together we have $\operatorname{rank}(L)=\operatorname{rank}(L')+\operatorname{rank}(\mathbb{1})=5N-1$. Details of the proof will be shown in an extended paper.

The matrix Ω codes the connectivities between different ratings in the movie-to-movie network, and could yield to a recommendation for a particular user. Since the matrix represents only the aggregated information, in order to obtain recommendations for a particular user, we need to utilize opinions expressed by this user. We do so by imposing these ratings as fixed elements of $\boldsymbol{h}$ in eq. (2). These fixed elements can be considered as a boundary condition of the given diffusion process; they influence our expectations on unexpressed ratings. In other words, large weights in Ω represent strong patterns in user ratings (*e.g.* most of those who rated movie X with 5 gave 3 to movie Y) and diffusion of the ratings expressed by a particular user in the movie-to-movie network makes use of these patterns.

The discussion above leads us to the equation

$$\Omega_i\boldsymbol{h}_i=\boldsymbol{h}_i, \tag{4}$$

where $\Omega_i:=\Omega$ for the rows corresponding to the movies unrated by user i and $\Omega_i:=\mathbb{1}$ for the remaining rows. Such a definition keeps entries corresponding to the movies rated by user i preserved. The solution of eq. (4) can be numerically obtained in a simple iterative way. We start with $\boldsymbol{h}_i^{(0)}$ where elements corresponding to the movies rated by user i are set according to these ratings and the remaining elements are set to zero. Then by the iteration equation $\boldsymbol{h}_i^{(n+1)}=\Omega_i\boldsymbol{h}_i^{(n)}$ we propagate already expressed opinions of user i over the network, eventually leading to the stationary solution $\boldsymbol{h}_i$. Intermediate results

[1]Here one can recall the famous set of equations for PageRank $G(i)$ of webpage i. It has the form $G(i)=\alpha+(1-\alpha)\sum_{j\sim i}G(j)/k_j$, where the subscript j runs over all the webpages that contain a link to webpage i $(j\sim i)$, for details see [13]. Here a similar scaling of the contributions by the inverse of the node degree arises. By a numerical solution of the set, one obtains values $G(i)$ which are essential for the Google search algorithm.

[2]Incidentally, the PageRank algorithm normalizes the flux outgoing from a node in a similar way and thus it also represents diffusion or a random walk. If one chooses the row normalization instead, the resulting process is equivalent to heat conduction in the network.

$\boldsymbol{h}_i^{(n)}$ contain information about the movies unrated by user i, which can give rise to a recommendation. We obtain the rating prediction as the standard weighted average. For example, if for a given movie in $\boldsymbol{h}_i$ we obtain the 5-tuple $(0.1, 0.2, 0.4, 0.3, 0.0)^T$, the rating prediction is $\hat{v} = 2.9$. Notice that if a user has rated no movies, we have to use a different method (for example the movie average introduced later) to make a prediction. This feature is common for recommender systems producing personalized predictions.

Avoiding the iterations. – While simple, the iterative way to solve eq. (4) has one important drawback: the iterations have to be made for every user separately. Consequently, the computational complexity of the algorithm is high. To get rid of this difficulty we rewrite eq. (4) as $L\boldsymbol{h}_i = \boldsymbol{j}_i$, again $L = 1 - \Omega$. Here the external flux $\boldsymbol{j}_i$ is non-zero only for the elements representing the boundary condition of user i.

The solution $\boldsymbol{h}_i$ can be formally written in the form $\boldsymbol{h}_i = \mathsf{G}\boldsymbol{j}_i$. This resembles the well-known Green function approach: once G is known, $\boldsymbol{h}_i$ can be found by a simple matrix multiplication. While the source term $\boldsymbol{j}_i$ is not *a priori* known, we can get rid of it by reshuffling of the movies and grouping the boundary elements in $\boldsymbol{h}_i$. After this formal manipulation we obtain

$$\begin{pmatrix} \boldsymbol{h}_i^{\mathrm{B}} \\ \boldsymbol{h}_i^{\mathrm{F}} \end{pmatrix} = \begin{pmatrix} \mathsf{G}_{\mathrm{BB}} & \mathsf{G}_{\mathrm{BF}} \\ \mathsf{G}_{\mathrm{FB}} & \mathsf{G}_{\mathrm{FF}} \end{pmatrix} \begin{pmatrix} \boldsymbol{j}_i^{\mathrm{B}} \\ \boldsymbol{0} \end{pmatrix}, \qquad (5)$$

where B stands for *boundary* and F for *free*. Now it follows that $\boldsymbol{h}_i^{\mathrm{B}} = \mathsf{G}_{\mathrm{BB}}\boldsymbol{j}_i^{\mathrm{B}}$ and $\boldsymbol{h}_i^{\mathrm{F}} = \mathsf{G}_{\mathrm{FB}}\boldsymbol{j}_i^{\mathrm{B}}$, leading us to the final result

$$\boldsymbol{h}_i^{\mathrm{F}} = \mathsf{G}_{\mathrm{FB}}\mathsf{G}_{\mathrm{BB}}^{-1}\boldsymbol{h}_i^{\mathrm{B}}. \qquad (6)$$

Since most users have rated only a small part of all M movies, the dimension of G_{BB} is usually much smaller than that of G and thus the inversion $\mathsf{G}_{\mathrm{BB}}^{-1}$ is cheap.

The last missing point is that since L is singular (as we have mentioned, $\mathrm{rank}(L) = 5N - 1$), the form of G cannot be obtained by inverting L. Hence we use the *Moore-Penrose pseudoinverse* [15]

$$\mathsf{G} = L^{\dagger} = \lim_{k\to\infty} \left[1 + \Omega + \Omega^2 + \cdots + \Omega^k - k\boldsymbol{w}_{\mathrm{R}}\boldsymbol{w}_{\mathrm{L}}\right], \qquad (7)$$

where $\boldsymbol{w}_{\mathrm{R}}$ and $\boldsymbol{w}_{\mathrm{L}}$ is the right and left eigenvector of Ω, respectively, both corresponding to the eigenvalue 1. For practical purposes, the infinite summation in eq. (7) can be truncated at a finite value k.

Personal polarization. – Before the described method can be used in real-life examples, there is one important technical problem. Each user has a different style of rating —some people tend to be very strict and on average give low marks, some people prefer to give either 1 or 5, some do not like to give low marks, and so forth. Thus, ratings cannot be grouped together in matrices $\mathsf{W}_{\alpha\beta}$ in the straightforward and naïve way we described before for they mean different things to different people.

To deal with this phenomenon, which we refer to as personal polarization, *unification* of ratings from different users is used before summing users' contributions in the object-to-object network. Consequently, before reporting resulting predictions to a user, the output of the algorithm has to be shifted back to the user's scale and *personalization* is needed.

To characterize the rating profile of user i we use the mean μ_i and the standard deviation σ_i of the votes given by him, and we compare these values with the mean m_i and the standard deviation s_i of the ratings given by all users. Notably, the quantities m_i and s_i take into account only the movies rated by user i —if a user has a low average rating because he has been rating only bad movies, there is no need to manipulate his ratings. To conform a user rating profile to the society rating profile we use the linear transformation

$$u_{i\alpha} = m_i + (v_{i\alpha} - \mu_i)\,\frac{s_i}{\sigma_i}. \qquad (8)$$

Personalization of the predicted value is done by the inverse formula $v_{i\alpha} = \mu_i + (u_{i\alpha} - m_i)\sigma_i/s_i$. We can notice that while $v_{i\alpha}$ is an integer value, $u_{i\alpha}$ is a real number. Nevetheless, one can obtain its vector representation in the straightforward way: *e.g.* $u = 3.7$ is modelled by the vector $(0, 0, 0.3, 0.7, 0)^T$; the weighted mean corresponding to this vector is equal to the input value 3.7.

Benchmark methods. – In correlation-based methods, rating correlations between users are quantified and utilized to obtain predictions. We present here one implementation of such a method, which serves as a benchmark for the proposed diffusion model. The correlation C_{ij} between users i and j is calculated with Pearson's formula

$$C_{ij} = \frac{\sum_{\alpha}^{*}(v_{i\alpha} - \mu_i)(v_{j\alpha} - \mu_j)}{\sqrt{\sum_{\alpha}^{*}(v_{i\alpha} - \mu_i)^2}\sqrt{\sum_{\alpha}^{*}(v_{j\alpha} - \mu_j)^2}}, \qquad (9)$$

where we sum over all movies rated by both i and j (to remind this, there is a star added to the summation symbols); $C_{ij} := 0$ when users i and j have no movies in common. Due to the data sparsity, the number of user pairs with zero correlation can be high and the resulting prediction performance poor. To deal with this effect, in [16] it is suggested to replace the zero correlations by the society average of C_{ij}. In the numerical tests presented in this letter the resulting improvement was small and thus we use eq. (9) in its original form. Finally, the predictions are obtained using the formula

$$\hat{v}_{i\alpha} = \mu_i + \sum\nolimits_j' \frac{C_{ij}}{\sum_k' C_{ik}}\,(v_{j\alpha} - \mu_j). \qquad (10)$$

Here we sum over the users who have rated movie α (prime symbols added to sums are used to indicate this), the term $\sum_k' C_{ik}$ serves as a normalization factor.

As a second benchmark method we use recommendation by the movie average (MA) where one has $\hat{v}_{i\alpha} = m_\alpha$, m_α

is the average rating of movie α. This method is not personalized (for a given object, all users obtain the same prediction) and has an inferior performance. As it is very fast and easy to implement, it is still widely used. Notably, when the unification-personalization scheme is employed together with MA, the predictions get personalized. As we will see later, in this way the prediction performance is increased considerably without a notable impact on the computation complexity.

Numerical results. – To test the proposed diffusion based (DB) method we use the GroupLens project data, available at `www.grouplens.org`. The total number of users is $M = 943$, the total number of movies is $N = 1682$, and the ratings are integer values from 1 to 5. The number of given ratings is 100000, corresponding to the voting matrix sparsity around 6%.

To test the described methods, randomly selected 10% of the available data is transfered to the probe file $\mathcal{P}$, and the remaining 90% is used as an input data for the recommendation. Then we make a prediction for all entries contained in the probe and measure the difference between the predicted value $\hat{v}_{i\alpha}$ and the actual value $v_{i\alpha}$. For an aggregate review of the prediction performance we use two common quantities: *root-mean-square error* (RMSE) and *mean absolute error* (MAE). They are defined as

$$\mathrm{MAE} = \frac{1}{n} \sum_{\mathcal{P}} |v_{i\alpha} - \hat{v}_{i\alpha}|, \tag{11a}$$

$$\mathrm{RMSE} = \left[\frac{1}{n} \sum_{\mathcal{P}} (v_{i\alpha} - \hat{v}_{i\alpha})^2 \right]^{1/2}, \tag{11b}$$

where the summations go over all user-movie pairs (i, α) included in the probe $\mathcal{P}$ and n is the number of these pairs in each probe dataset. To obtain a better statistics, the described procedure can be repeated many times with different selections of the probe data. We used 10 repetitions and in addition to the averages of MAE and RMSE we found also standard deviations of both quantities.

In contrast with the expectations, in fig. 2 it can be seen that the prediction performance is getting worse by a small amount when more than one iteration of eq. (4) is used to obtain the prediction. Probably this is due to the presence of overfitting —starting from the second iteration, our expectations are influenced not only by actually expressed ratings but also by our expectations about unexpressed ratings obtained in previous iteration steps. Nevertheless, as will be shown later, the performance achieved by the first iteration is good and justifies the validity of the proposed model. In the following paragraphs we use only one iteration to obtain the predictions. Consequently, the Green function method introduced above is not necessary —we decided to expose it in this paper because it can be useful with other datasets.

In table 1 we compare the prediction accuracy for the movie-average method (MA), the correlation-based method (CB), and for the opinion diffusion (OD). To measure the prediction performances we use both RMSE

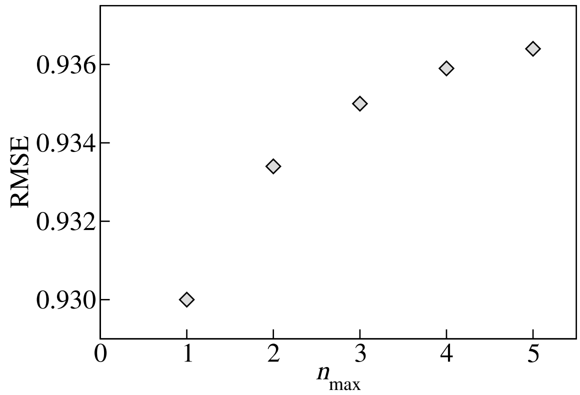

Fig. 2: Prediction performance for the predictions $\hat{v}_{i\alpha}$ obtained by iterations of eq. (4) using various numbers of iterations steps.

Table 1: Comparison of the three recommendation methods: movie average (MA), correlation-based method (CB), and opinion diffusion (OD). Presented values are averages obtained using 10 different probes; standard deviations are approximately 0.01 in all investigated cases.

	No unification		With unification	
Method	RMSE	MAE	RMSE	MAE
MA	1.18	0.91	1.01	0.79
CB	1.09	0.86	1.09	0.86
OD	1.00	0.80	0.93	0.73

and MAE as defined above. All three methods are tested both with and without employing the unification-personalization scheme. In accordance with expectations, for MA and OD the performances with unification included are better than without it; for the simplest tested method, MA, the difference is particularly remarkable. By contrast, CB is little sensitive to the unification procedure and when we drop the multiplication by σ_i/s_i from the unification-personalization process given by eq. (8), the difference disappears completely (which can be also confirmed analytically). According to the prediction performances shown in table 1 we can conclude that the diffusion method outperforms the other two clearly in all tested cases (RMSE/MAE, with/without unification). When computation complexity is taken into account, it can be shown that if $M > N$, the proposed method is more effective than correlation-based methods (but, of course, less effective than using the movie average).

Conclusion. – We have proposed a novel recommendation method based on diffusion of opinions expressed by a user over the object-to-object network. Since the rating polarization effect is present, we have suggested the unification-personalization approach as an additional layer of the recommender system. To allow a computation reduction with some datasets, a Green function method has been introduced. The proposed method has been compared with two standard recommendation algorithms and it has achieved consistently better results. Notably, it is executable even for the large dataset (17770 movies, 480189 users) released by Netflix (a DVD rental company, see `www.netflixprize.com`). In addition, our model is tune-free in essence —it does not require extensive testing and optimization to produce a high-quality output. This is good news for practitioners.

* * *

This work is partially supported by the Swiss National Science Foundation (project 205120-113842). We acknowledge SBF (Switzerland) for financial support through project C05.0148 (Physics of Risk), TZ acknowledges NNSFC (No. 10635040). We kindly acknowledge the computing resources provided by Swiss National Supercomputing Center.

REFERENCES

[1] Faloutsos M., Faloutsos P. and Faloutsos C., *Comput. Commun. Rev.*, **29** (1999) 251.
[2] Broder A., Kumar R., Maghoul F., Raghavan P., Rajagopalan S., Stata R., Tomkins A. and Wiener J., *Comput. Netw.*, **33** (2000) 309.
[3] Balabanović M. and Shoham Y., *Commun. ACM*, **40** (1997) 66.
[4] Pazzani M. J., *Artif. Intell. Rev.*, **13** (1999) 393.
[5] Maslov S. and Zhang Y.-C., *Phys. Rev. Lett.*, **87** (2001) 248701.
[6] Herlocker J. L., Konstan J. A., Terveen L. G. and Riedl J. T., *ACM Trans. Inf. Syst.*, **22** (2004) 5.
[7] Adomavicius G. and Tuzhilin A., *IEEE Trans. Knowl. Data Eng.*, **17** (2005) 734.
[8] Newman M. E. J., *SIAM Rev.*, **45** (2003) 167.
[9] Zhou T., Ren J., Medo M. and Zhang Y.-C., *Phys. Rev. E*, **76** (2007) 046115, arXiv:0707.0540.
[10] Zhang Y.-C., Blattner M. and Yu Y.-K., *Phys. Rev. Lett.*, **99** (2007) 154301.
[11] Percival I., *Quantum State Diffusion* (Cambridge University Press) 1999.
[12] Kim B. J., Hong H. and Choi M. Y., *Phys. Rev. B*, **68** (2003) 014304.
[13] Brin S. and Page L., *Comput. Netw. ISDN Syst.*, **30** (1998) 107.
[14] Ou Q., Jin Y.-D., Zhou T., Wang B.-H. and Yin B.-Q., *Phys. Rev. E*, **75** (2007) 021102.
[15] Penrose R. A., *Proc. Cambridge Philos. Soc.*, **51** (1955) 406.
[16] Blattner M., Laureti P. and Huntziker A., arXiv:0709.2562.