MF models: Notes

Aaron Tuor

September, 2015

Contents

1 Notation and Definitions

Sets are denoted with capital script letters.
Matrices are denoted with capital letters.
The entry of a matrix A at row i and column j is denoted A_{ij}
For an $m \times n$ matrix A the i-th row is denoted $\underbrace{A_{i:}}_{1\times n}$ and the j-th column is denoted $\underbrace{A_{:j}}_{m\times 1}$.
To eliminate parentheses, subscripts have precedence over all operators, e.g.,
$\underbrace{A_{i:}^{\mathsf{T}}}_{n\times 1} = \underbrace{(A_{i:})^{\mathsf{T}}}_{n\times 1}$, for an $m \times n$ matrix A.
$A \circ B$ is the elementwise product (Hadamard product) of A and B, i.e.,

$$(A \circ B)_{ij} = A_{ij}B_{ij}.$$

$\|A\|_F = \sqrt{\sum_i \sum_j A_{ij}^2}$ is the Frobenius norm of A.

Special ranges of indexing letters are reserved for distinguishing between users, items, words, and latent factors:

Entity	Items	Factors	Users	Words
Indices	h, i, j	p, q, r	u, v, w	x, y, z

Definitions

R denotes the ***user-item utility matrix*** or simply the ***utility matrix*** where

$$R_{ui} = \begin{cases} \text{User u's rating of item i} & \text{for observed ratings,} \\ 0 & \text{for unknown ratings.} \end{cases}$$

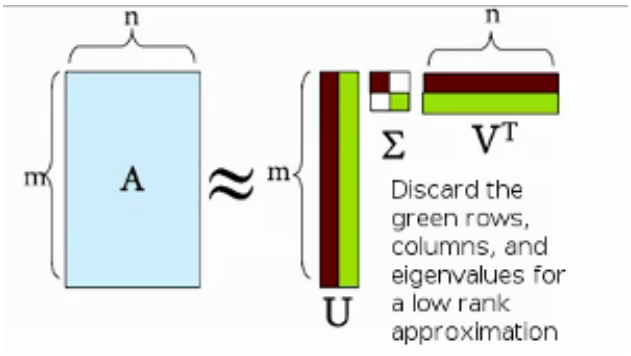
$\mathcal{K} = \{(u, i) \mid R_{ui} \text{ is known}\}$.
$\mathcal{K}_u = \{i \mid R_{ui} \text{ is known }\}$.
Assuming R is $m \times n$, Ω is an $m \times n$ matrix such that:

$$\Omega_{ui} = \begin{cases} 1 & \text{if } (u, i) \in \mathcal{K} \\ 0 & \text{otherwise} \end{cases}$$

2 MF Models

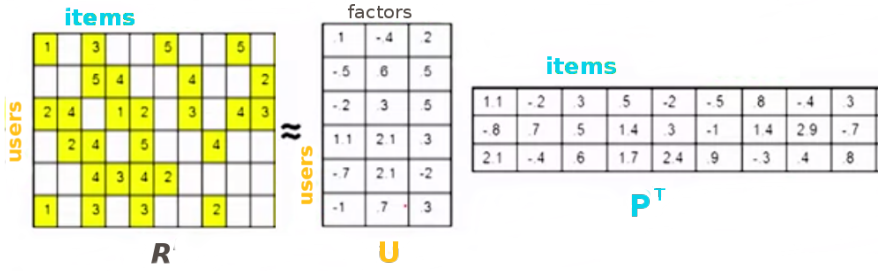
The basic idea behind matrix factorization models is that the information encoded for items in the columns of the utility matrix, and for users in the rows of the utility matrix is not exactly independent. Users and items should be related in some way that may be obscured by the noisiness and sparsity of the data. That is, if we did happen to have ratings for all the users and all the items in the dataset we should find either the rank of the utility matrix was significantly lower, perhaps orders of magnitude, lower than either m or n, or else given

a singular value decomposition of the complete utility matrix, $R = U\Sigma V^{\mathsf{T}}$, by zeroing out the k last diagonal entries of Σ we are left with a close approximation of the original ratings matrix.



3 Latent Factors

Assume $\underbrace{R}_{m\times n}$ can be factored into two matrices, $\underbrace{U}_{m\times k}$ and $\underbrace{P^{\mathsf{T}}}_{k\times n}$, so that $\underbrace{U}_{m\times k}\underbrace{P^{\mathsf{T}}}_{k\times n} \approx \underbrace{R}_{m\times n}$. One can imagine the eigenvalues contained in the matrix Σ of the SVD of R as having been arbitrarily absorbed into either U or P^T or both. U_{vr} is interpreted as user v's level of appreciation of the latent or hidden factor r. P_{ir} is interpreted as item i's level of possession of latent factor r. $U_{v:}$ is called the user profile vector as it contains user v's preferences for the k latent factors. $P_{i:}$ is called the item profile vector as it contains item i's level of possession of latent factors. User-item interaction between user v and item i is modeled by $\underbrace{U_{v:}}_{1\times k}\underbrace{P_{i:}^{\mathsf{T}}}_{k\times 1}$.



A natural loss function for training this model is:

$$\mathcal{L}(U,P) = \frac{1}{2}\sum_{(v,i)\in\mathcal{K}}(R_{vi} - U_{v:}P_{i:}^{\mathsf{T}})^2 = \frac{1}{2}\|\Omega\circ(R - UP^{\mathsf{T}})\|_F^2 \tag{3.1}$$

[4]

This model is often referred to in the literature as SVD.

To derive the gradients with respect to matrices U and P^T we'll use the gradient identities derived in section 8.

By (8.17) we have that:

$$\nabla_U \mathcal{L} = \nabla_U \frac{1}{2}\|\Omega \circ (R - UP^\mathsf{T})\|_F^2 = -2 * \frac{1}{2}\Big(\Omega \circ \Omega \circ \big(R - UP^\mathsf{T}\big)\Big)P = -\Big(\Omega \circ \big(R - UP^\mathsf{T}\big)\Big)P \quad (3.2)$$

Notice that,

$$\mathcal{L}(U, P) = \frac{1}{2}\|\Omega \circ (R - UP^\mathsf{T})\|_F^2 = \frac{1}{2}\|\Omega^\mathsf{T} \circ (R^\mathsf{T} - PU^\mathsf{T})\|_F^2. \quad (3.3)$$

So,

$$\nabla_P \mathcal{L} = -\Big(\Omega^\mathsf{T} \circ \big(R^\mathsf{T} - PU^\mathsf{T}\big)\Big)U. \quad (3.4)$$

3.1 L_2 regularization

With so many parameters $(k * m + k * n)$ this model may be prone to over fitting. Let λ_1 be a hyperparameter controlling the amount of regularization. A reasonable loss function for this model with L_2 regularization is:

$$\mathcal{L}_2(U, P) = \frac{1}{2}\|\Omega \circ (R - UP^\mathsf{T})\|_F^2 + \frac{1}{2}\lambda_1(\|U\|_F^2 + \|P\|_F^2) \quad (3.5)$$

[4]

The gradients:

$$\begin{gathered} \nabla_U \mathcal{L}_2 = \nabla_U \Big(\frac{1}{2}\|\Omega \circ (R - UP^\mathsf{T})\|_F^2 + \frac{1}{2}\lambda_1(\|U\|_F^2 + \|P\|_F^2)\Big) = \\ \nabla_U \mathcal{L} + \nabla_U \frac{1}{2}\lambda_1(\|U\|_F^2 + \|P\|_F^2) = \nabla_U \mathcal{L} + \frac{1}{2}\lambda_1 \nabla_U \|U\|_F^2 = \nabla_U \mathcal{L} + \lambda_1 U. \end{gathered} \quad (3.6)$$

By a similar derivation,

$$\nabla_P \mathcal{L}_2 = \nabla_P \mathcal{L} + \lambda_1 P. \quad (3.7)$$

3.2 Baseline Predictors

From Koren and Bell. *Advances in Collaborative Filtering* [4]:

> CF models try to capture the interactions between users and items that produce the different rating values. However, much of the observed rating values are due to effects associated with either users or items, independently of their interaction. A principal example is that typical CF data exhibit large user and item biases i.e., systematic tendencies for some users to give higher ratings than others, and for some items to receive higher ratings than others. We will encapsulate those

effects, which do not involve user-item interaction, within the baseline predictors (also known as biases). Because these predictors tend to capture much of the observed signal, it is vital to model them accurately. Such modeling enables isolating the part of the signal that truly represents user-item interaction, and subjecting it to more appropriate user preference models. Denote by μ the overall average rating. A baseline prediction for an unknown rating R_{ui} is denoted by b_{ui} and accounts for the user and item effects:

$$b_{ui} = \mu + a_u + b_i \tag{3.8}$$

The parameters a_u and b_i indicate the observed deviations of user u and item i, respectively, from the average.

Incorporating user and item biases into the MF model gives the a rating prediction of:

$$R_{vi} \approx \mu + a_v + b_i + U_{v:}P_{i:}^{\mathsf{T}}. \tag{3.9}$$

The corresponding loss function with L_2 regularization is [4]:

$$\mathcal{L}_3 = \frac{1}{2}\sum_{(v,i)\in\mathcal{K}} (R_{vi} - (\mu + a_v + b_i + U_{v:}P_{i:})^{\mathsf{T}})^2 + \frac{1}{2}\lambda_1(\|U_{v:}\|^2 + \|P_{i:}\|^2 + a_u^2 + b_i^2). \tag{3.10}$$

$\mathcal{L}_3$ can be written in matrix form in terms of the matrices $\hat{R}$, $\hat{U}$, and $\hat{P}$ where:

$$\hat{R}_{vi} = R_{vi} - \mu, \qquad \hat{U} = \begin{bmatrix} U_{11} & \dots & U_{1k} & a_1 & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ U_{m1} & \dots & U_{mk} & a_m & 1 \end{bmatrix}, \quad \text{and} \quad \hat{P}^{\mathsf{T}} = \begin{bmatrix} P_{11} & \dots & P_{n1} \\ \vdots & \ddots & \vdots \\ P_{1k} & \dots & P_{nk} \\ 1 & \dots & 1 \\ b_1 & \dots & b_n \end{bmatrix}$$

so that

$$\hat{R}_{vi} \approx \hat{U}\hat{P}^{\mathsf{T}}. \tag{3.11}$$

Since the parameter matrices have now been infiltrated by constants our gradient formula doesn't exactly work anymore. This can be resolved by defining masks, Ω_X, where:

$$\Omega_{Xij} = \begin{cases} 0 & \text{if } X_{ij} \text{ is a constant,} \\ 1 & \text{otherwise.} \end{cases}$$

So the loss and associated gradients are now:

$$\mathcal{L}_3 = \frac{1}{2}\|\Omega \circ (\hat{R} - \hat{U}\hat{P}^{\mathsf{T}})\|_F^2 + \frac{1}{2}\lambda_1(\|\hat{U}\|_F^2 + \|\hat{P}\|_F^2) \tag{3.12}$$

$$\nabla_{\tilde{U}}\mathcal{L}_3 = -\Omega_{\hat{U}} \circ \Big(\big(\Omega \circ (\hat{R} - \hat{U}\hat{P}^{\mathsf{T}})\big)\hat{P} + \lambda_1\hat{U}\Big) \tag{3.13}$$

$$\nabla_{\tilde{P}}\mathcal{L}_3 = -\Omega_{\hat{P}} \circ \Big(\big(\Omega \circ (\hat{R} - \hat{U}\hat{P}^{\mathsf{T}})\big)\hat{P} + \lambda_1 \hat{P} \Big) \tag{3.14}$$

Koren and Bell report evaluating this model on the Netflix data with a learning rate of 0.005 and regularization weight of 0.02.

Koren and Bell remark that better results are obtained by tuning separate regularization weights for each type of parameter (user bias, item bias, user profiles, and item profiles). This can be written in matrix form in terms of two weight matrices:

$$\Lambda_1 = \underbrace{\begin{bmatrix} \lambda_1 & \dots & \lambda_1 & \lambda_3 & 0 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \lambda_1 & \dots & \lambda_1 & \lambda_3 & 0 \end{bmatrix}}_{m \times (k+2)}, \text{ and } \Lambda_2 = \underbrace{\begin{bmatrix} \lambda_2 & \dots & \lambda_2 & 0 & \lambda_4 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ \lambda_2 & \dots & \lambda_2 & 0 & \lambda_4 \end{bmatrix}}_{n \times (k+2)}$$

where λ_1^2 is the regularization weight for user profiles, λ_3^2 is the regularization weight for user bias, λ_2^2 is the regularization weight for item profiles, and λ_4^2 is the regularization weight for item bias. The loss function with separate regularization weights is now:

$$\mathcal{L}_4 = \frac{1}{2} \sum_{(v,i) \in \mathcal{K}} (R_{vi} - (\mu + a_v + b_i + U_{v:}P_{i:})^{\mathsf{T}})^2 + \frac{1}{2}(\lambda_1^2 \|U_{v:}\|^2 + \lambda_2^2 \|P_{i:}\|^2 + \lambda_3^2 a_u^2 + \lambda_4^2 b_i^2) \tag{3.15}$$

$$= \frac{1}{2}\|\Omega \circ (\hat{R} - \hat{U}\hat{P}^{\mathsf{T}})\|_F^2 + \frac{1}{2}(\|\Lambda_1 \circ \hat{U}\|_F^2 + \|\Lambda_2 \circ \hat{P}\|_F^2) \tag{3.16}$$

$$\Rightarrow \nabla_{\tilde{U}}\mathcal{L}_4 = -\big(\Omega \circ (\hat{R} - \hat{U}\hat{P}^{\mathsf{T}})\big)\hat{P} + \Lambda_1 \circ \Lambda_1 \circ \hat{U} \tag{3.17}$$

$$\nabla_{\tilde{P}}\mathcal{L}_4 = -\big(\Omega \circ (\hat{R} - \hat{U}\hat{P}^{\mathsf{T}})\big)\hat{P} + \Lambda_2 \circ \Lambda_2 \circ \hat{P}. \tag{3.18}$$

Koren and Bell [4] also suggest that applying different learning rates for gradient descent on the different types of parameters will improve prediction accuracy. A scheme for implementing this training method is outlined in [**?**].

4 SVD++ [4]

SVD++ incorporates implicit feedback into the MF model. For r sources of implicit feedback and k latent factors, the prediction rule for SVD++ is:

$$R_{vi} \approx \mu + a_v + b_i + \Big(\underbrace{U_{v:}}_{1\times k} + |\mathcal{N}^1(v)|^{-\frac{1}{2}} \sum_{j \in \mathcal{N}^1(v)} \underbrace{y_j^{(1)}}_{1\times k} + \dots + |\mathcal{N}^r(v)|^{-\frac{1}{2}} \sum_{j \in \mathcal{N}^r(v)} \underbrace{y_j^{(r)}}_{1\times k} \Big) P_{i:}^{\mathsf{T}} \tag{4.1}$$

Here $\mathcal{N}^p(v)$ is the set of items for which user v has given implicit feedback of type p. For each item j, y_j is a row vector with an entry for each latent factor. Koren and Bell mention that "a significant signal can be captured by accounting for which items users rate, regardless of

their rating value". If we let $\mathcal{R}(v)$ be the set of items which a user has rated then an SVD++ prediction that incorporates the implicit feedback of which items a user rates is:

$$R_{vi} \approx \mu + a_v + b_i + \Big(U_{v:} + |\mathcal{R}(v)|^{-\frac{1}{2}} \sum_{j \in \mathcal{R}(v)} y_j\Big) P_{i:}^{\mathsf{T}} \tag{4.2}$$

To express this prediction rule and associated loss function in matrix form define the matrices:

$$\hat{Y} = \underbrace{\begin{bmatrix} y_1 & 0 & 0 \\ \vdots & \vdots & \vdots \\ y_n & 0 & 0 \end{bmatrix}}_{k+2} \qquad C = \left.\begin{bmatrix} |\mathcal{R}(1)|^{-\frac{1}{2}} & 0 & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & |\mathcal{R}(m)|^{-\frac{1}{2}} \end{bmatrix}\right\} m$$

We now have the following equality:

$$R_{vi} - \Big(\mu + a_v + b_i + \Big(U_{v:} + |\mathcal{R}(v)|^{-\frac{1}{2}} \sum_{j \in \mathcal{R}(v)} y_j\Big) P_{i:}^{\mathsf{T}}\Big) = \Big(\underbrace{\hat{R}}_{m \times n} - \big(\underbrace{\hat{U}}_{m \times k+2} + \underbrace{C}_{m \times m} \underbrace{\Omega}_{m \times n} \underbrace{\hat{Y}}_{n \times k+2}\big) \underbrace{\hat{P}^{\mathsf{T}}}_{k+2 \times n}\Big)_{vi} \tag{4.3}$$

So the loss function without regularization is (to implement without bias replace all matrices with tildes by their unadorned counterparts):

$$\mathcal{L}_5 = \textbf{(i)} \frac{1}{2} \|\Omega \circ \Big(\hat{R} - \big(\hat{U} + C\Omega\hat{Y}\big)\hat{P}^{\mathsf{T}}\Big)\|_F^2 = \textbf{(ii)} \frac{1}{2} \|\Omega \circ \Big(\big(\hat{R} - C\Omega\hat{Y}\hat{P}^{\mathsf{T}}\big) - \hat{U}\hat{P}^{\mathsf{T}}\Big)\|_F^2 =$$
$$\textbf{(iii)} \frac{1}{2} \|\Omega^{\mathsf{T}} \circ \Big(\big(\hat{R}^{\mathsf{T}} - \hat{P}\big(\hat{U}^{\mathsf{T}} + \hat{Y}^{\mathsf{T}}\Omega^{\mathsf{T}}C^{\mathsf{T}}\big)\big)\Big)\|_F^2 = \textbf{(iv)} \frac{1}{2} \|\Omega \circ \Big(\big(\hat{R} - \hat{U}\hat{P}^{\mathsf{T}}\big) - C\Omega\hat{Y}\hat{P}^{\mathsf{T}}\Big)\|_F^2 \tag{4.4}$$

From (8.17) and (ii):

$$\nabla_{\hat{U}} \mathcal{L}_5 = -\Omega_{\hat{U}} \circ \Big(\Big(\Omega \circ \Big(\big(\hat{R} - C\Omega\hat{Y}\hat{P}^{\mathsf{T}}\big) - \hat{U}\hat{P}^{\mathsf{T}}\Big)\Big)\hat{P}\Big) \tag{4.5}$$

From (8.17) and (iii):

$$\nabla_{\hat{P}} \mathcal{L}_5 = -\Omega_{\hat{P}} \circ \Big(\Big(\Omega^{\mathsf{T}} \circ \Big(\hat{R}^{\mathsf{T}} - \hat{P}\big(\hat{U}^{\mathsf{T}} + \hat{Y}^{\mathsf{T}}\Omega^{\mathsf{T}}C^{\mathsf{T}}\big)\Big)\Big)\big(\hat{U} + C\Omega\hat{Y}\big)\Big) \tag{4.6}$$

From (8.1) and (iv):

$$\nabla_{\hat{Y}} \mathcal{L}_5 = -\Omega_{\hat{U}} \circ \Big(\Omega^{\mathsf{T}} C^{\mathsf{T}} \Big(\Omega \circ \Big(\big(\hat{R} - \hat{U}\hat{P}^{\mathsf{T}}\big) - C\Omega\hat{Y}\hat{P}^{\mathsf{T}}\Big)\Big)P\Big) \tag{4.7}$$

5 Joint Factorization (JF) Models

Joint Factorization Models are another method used to incorporate data besides ratings into a MF approach. The idea is to tie a matrix factorization involving ratings to another factorization involving some other form of data. One source of such data which has been

exploited using JF models is text associated with items. This sort of approach has been shown to be successful at addressing the cold start problem (recommending new items which haven't been rated to users). Suppose document j is a collection of texts associated with item j. The texts contain words from a dictionary of length v. Then $Q_{xj} = f(x, j)$ where f may be defined in the following three ways:

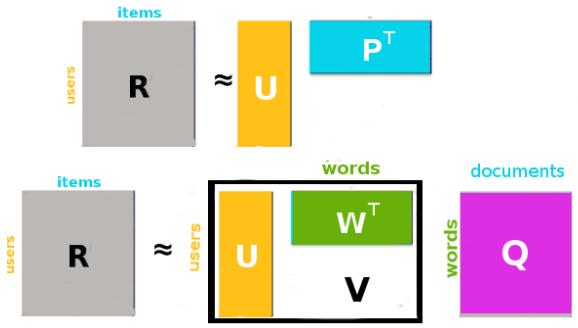
(i) $f(x,j) = \begin{cases} 1 & \text{if word x occurs in document j,} \\ 0 & \text{otherwise.} \end{cases}$

(ii) $f(x, j) =$ the number of occurrences of word x in document j.

(iii) $f(x, j) = \textbf{TF-IDF}(x, j)$

Now we can assume that words can be represented in a lower dimensional space than the n-dimensional document space; the same space that user and item profiles belong to. So now, we learn the item profile vectors of P and user profile vectors in the matrix U, as well as word profile vectors in the matrix W.

5.1 Brian's Model



W_{xr} is now interpeted as word x's association with latent factor r. $V_{ux} = U_{:u}W_{:x}^{\mathsf{T}}$ is then a user word prediction (how much a user likes a certain word). The prediction function is

$$R_{ui} \approx V_{u:}Q_{:i}. \tag{5.1}$$

A new item may be recommended by appending its document term vector to Q. Also, the refinements from any of the preceding models can be incorporated into this model without much trouble. Let $\widetilde{\mathcal{L}}_p$ be the loss function from any of the models in previous sections without regularization terms, $\mathcal{F}_p$ be the associated regularization terms, λ be a parameter determining weights on the factorizations, and λ_7 be a weight controlling regularization of W. We can add a word bias to the model by appending two columns to W so that:

$$\hat{W}^{\mathsf{T}} = \begin{bmatrix} W_{11} & \dots & W_{n1} \\ \vdots & \ddots & \vdots \\ W_{1k} & \dots & W_{nk} \\ 1 & \dots & 1 \\ c_1 & \dots & c_n \end{bmatrix}$$

Below is the schema to incorporate any of the previous models into this JF model. (for models incorporating bias, replace U, W, and P with $\hat{U}, \hat{W}$, and $\hat{P}$ and elementwise multiply appropriate masks to gradient formulas). The loss and gradients:

$$\mathcal{L}_7 = \frac{1}{2}\lambda\|\Omega \circ \big(R - UW^T Q\big)\|_F^2 + (1-\lambda)\widetilde{\mathcal{L}}_p + \mathcal{F}_p + \frac{1}{2}\lambda_7\|W\|_F^2 \tag{5.2}$$

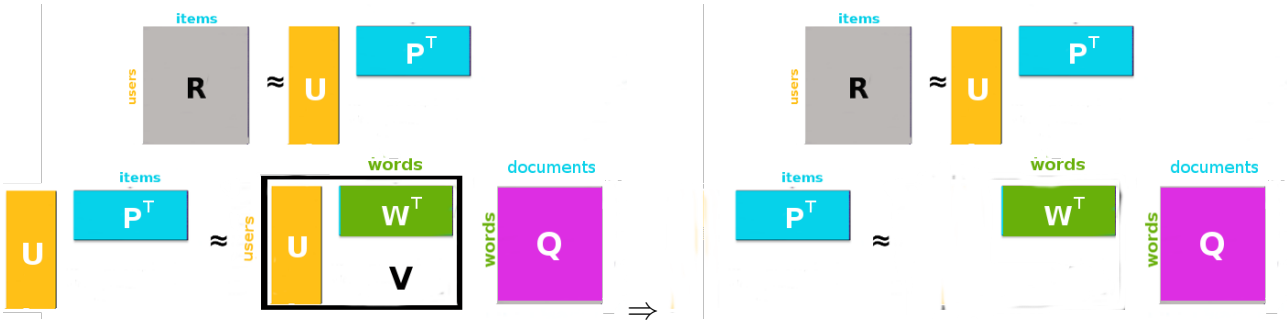
$$\nabla_U \mathcal{L}_7 = -\lambda\Omega \circ (R - UW^\mathsf{T} Q)Q^\mathsf{T} W + (\lambda - 1)\nabla_U \widetilde{\mathcal{L}}_p + \nabla_U \mathcal{F}_p \tag{5.3}$$

$$\nabla_P \mathcal{L}_7 = (\lambda - 1)\nabla_P \widetilde{\mathcal{L}}_p + \nabla_P \mathcal{F}_p \tag{5.4}$$

$$\nabla_W \mathcal{L}_7 = -\lambda Q\Big(\Omega^\mathsf{T} \circ \big(R^\mathsf{T} - Q^\mathsf{T} W U^\mathsf{T}\big)\Big)U + \lambda_7 W \tag{5.5}$$

5.2 Eliminating Ω from the second factorization

To avoid a loss of information in learning the parameters contained in the W matrix we can eliminate the mask Ω from the bottom factorization in the previous model by replacing R with UP^T in the bottom factorization and cancelling U on both sides of the equation. An added bonus is that this model is more computationally efficient. The prediction rule remains the same.



The loss and gradients:

$$\mathcal{L}_8 = \frac{1}{2}\lambda\|P^\mathsf{T} - W^\mathsf{T} Q\|_F^2 + (1-\lambda)\widetilde{\mathcal{L}}_p + \mathcal{F}_p + \frac{1}{2}\lambda_7\|W\|_F^2 \tag{5.6}$$

$$\nabla_U \mathcal{L}_8 = (\lambda - 1)\nabla_U \widetilde{\mathcal{L}}_p + \nabla_U \mathcal{F}_p \tag{5.7}$$

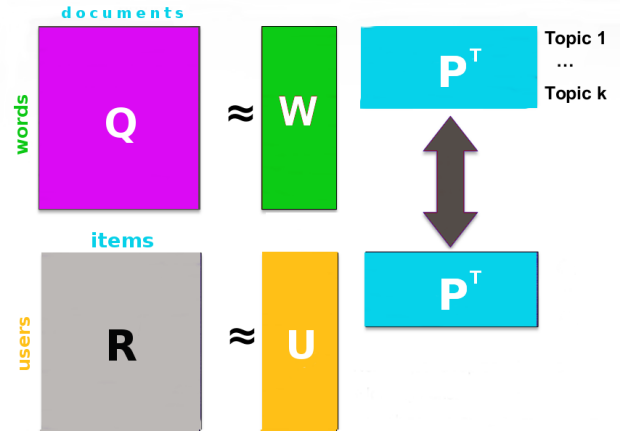
$$\nabla_P \mathcal{L}_8 = -\lambda(Q^\mathsf{T} W - P) + (\lambda - 1)\nabla_P \widetilde{\mathcal{L}}_p + \nabla_P \mathcal{F}_p \tag{5.8}$$

$$\nabla_W \mathcal{L}_8 = -\lambda Q(P - Q^\mathsf{T} W) + \lambda_7 W \tag{5.9}$$

5.3 Local Collective Embeddings

A different approach to incorporating the same information contained in text is the Local Collective Embeddings model. The idea is that the dot product of a word profile vector and item profile vector can predict how often or important a word is to a particular document associated with an item. A decision rule for a new item i is to derive a item profile

from the least squares problem $WP_{:i} \approx Q_i$, and then use the standard prediction formula $R_{vi} = U_{v:}P_{i:}^{\mathsf{T}}$. Once again the refinements from previous models can easily be incorporated. The disadvantage to this model is that each time a new item is incorporated into the system a least squares problem must be solved to find the new item profile, whereas in the previous JF models it was only necessary to take the dot product of two vectors, a simpler computation.



The associated loss function and gradients:

$$\mathcal{L}_9 = \frac{1}{2}\lambda\|Q - WP^{\mathsf{T}}\|_F^2 + (1-\lambda)\widetilde{\mathcal{L}}_p + \mathcal{F}_p + \frac{1}{2}\lambda_7\|W\|_F^2 \tag{5.10}$$

$$\nabla_U \mathcal{L}_9 = (\lambda - 1)\nabla_U \widetilde{\mathcal{L}}_p + \nabla_U \mathcal{F}_p \tag{5.11}$$

$$\nabla_P \mathcal{L}_9 = -\lambda(Q^T - PW^{\mathsf{T}})W + (\lambda - 1)\nabla_P \widetilde{\mathcal{L}}_p + \nabla_P \mathcal{F}_p \tag{5.12}$$

$$\nabla_W \mathcal{L}_9 = -\lambda(Q - WP^{\mathsf{T}})P + \lambda_7 W \tag{5.13}$$

A more general discussion of JF approaches to Machine Learning can be found in "Relational learning via collective matrix factorization" by Singh and Gordon (2008) [1].

6 Tensor Decomposition

7 Non-linearity

8 General SVD Loss Gradient

$$\nabla_X \|D \circ (A \pm BXC)\|_F^2 = \pm 2B^\mathsf{T}\big(D \circ D \circ (A \pm BXC)\big)C^\mathsf{T} \tag{8.1}$$

Proof. To show the identity we'll find the expression for an arbitrary element of this gradient.

$$\frac{\partial}{\partial X_{kl}} \|\underbrace{D}_{m\times n} \circ (\underbrace{A}_{m\times n} \pm \underbrace{B}_{m\times p}\underbrace{X}_{p\times q}\underbrace{C}_{q\times n})\|_F^2 = \tag{8.2}$$

$$\frac{\partial}{\partial X_{kl}} \sum_{i=1}^{m}\sum_{j=1}^{n}\big(D_{ij}(A_{ij} \pm B_{i:}(XC)_{:j})\big)^2 = \tag{8.3}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{\partial}{\partial X_{kl}}\big(D_{ij}(A_{ij} \pm B_{i:}(XC)_{:j})\big)^2 = \tag{8.4}$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n}D_{ij}^2\frac{\partial}{\partial X_{kl}}\big(A_{ij} \pm B_{i:}(XC)_{:j}\big)^2 = \tag{8.5}$$

$$\pm 2\sum_{i=1}^{m}\sum_{j=1}^{n}\Big(D_{ij}^2\big(A_{ij} \pm B_{i:}(XC)_{:j}\big)\frac{\partial}{\partial X_{kl}}B_{i:}(XC)_{:j}\Big) = \tag{8.6}$$

$$\pm 2\sum_{i=1}^{m}\sum_{j=1}^{n}\Big(D_{ij}^2\big(A_{ij} \pm B_{i:}(XC)_{:j}\big)\frac{\partial}{\partial X_{kl}}\underbrace{\sum_{r=1}^{p}B_{ir}\sum_{v=1}^{q}X_{rv}C_{vj}}_{\textbf{see (8.15)}}\Big) = \tag{8.7}$$

$$\pm 2\sum_{i=1}^{m}\sum_{j=1}^{n}\Big(D_{ij}^2\big(A_{ij} \pm B_{i:}(XC)_{:j}\big)\sum_{r=1}^{p}B_{ir}\sum_{v=1}^{q}\frac{\partial}{\partial X_{kl}}X_{rv}C_{vj}\Big) = \tag{8.8}$$

$$\pm 2\sum_{i=1}^{m}\sum_{j=1}^{n}\Big(D_{ij}^2\big(A_{ij} \pm B_{i:}(XC)_{:j}\big)B_{ik}\frac{\partial}{\partial X_{kl}}X_{kl}C_{lj}\Big) = \tag{8.9}$$

$$\pm 2\sum_{i=1}^{m}\sum_{j=1}^{n}\Big(D_{ij}^2\big(A_{ij} \pm B_{i:}(XC)_{:j}\big)B_{ik}C_{lj}\Big) = \tag{8.10}$$

$$\pm 2\sum_{i=1}^{m}(B^\mathsf{T})_{ki}\Big((D \circ D)_{i:} \circ (A_{i:} \pm B_{i:}XC)\Big)(C^\mathsf{T})_{:l} = \tag{8.11}$$

$$\pm 2(B^\mathsf{T})_{k:}\Big((D \circ D) \circ (A \pm BXC)\Big)(C^\mathsf{T})_{:l} = \tag{8.12}$$

$$\pm 2\Big(B^\mathsf{T}\big(D \circ D \circ (A \pm BXC)\big)C^T\Big)_{kl} \tag{8.13}$$

$$\Rightarrow \pm 2\underbrace{B^\mathsf{T}}_{p\times m}\underbrace{\big(D \circ D \circ (A \pm BXC)\big)}_{m\times n}\underbrace{C^\mathsf{T}}_{n\times q} = \underbrace{\nabla_X \|D \circ (A \pm BXC)\|_F^2}_{p\times q} \tag{8.14}$$

□

$$\underbrace{B}_{m\times p}\underbrace{X}_{p\times q}\underbrace{C}_{q\times n}$$

$$B_{i:}(XC)_{:j} = B_{i:}\begin{bmatrix} X_{1:}C_{:j} \\ X_{1:}C_{:j} \\ \vdots \\ X_{1:}C_{:j} \end{bmatrix} = B_{i:}\begin{bmatrix} \sum_{v=1}^{q} X_{1v}C_{vj} \\ \vdots \\ \sum_{v=1}^{q} X_{pv}C_{vj} \end{bmatrix} = \sum_{r=1}^{p} B_{ir}\sum_{v=1}^{q} X_{rv}C_{vj} \tag{8.15}$$

Notice that (1) has almost the same form as (119) from *The Matrix Cookbook*. [8]

(119) $$\frac{\partial}{\partial X}\mathbf{Tr}\big[(AXB+C)(AXB+C)^{\mathsf{T}}\big] = 2B^{\mathsf{T}}(AXB+C)C^{\mathsf{T}} = \frac{\partial}{\partial X}\|C+AXB\|_F^2 \tag{8.16}$$

So, probably there is some really nice way to derive (8.1) using (119). In any case there are lots of useful identities we can derive from special cases of (8.1), including (119).

$$\nabla_X\|D\circ(A\pm XC)\|_F^2 = \pm 2(D\circ D\circ(A\pm XC))C^T \tag{8.17}$$

$$\nabla_X\|A\pm XC\|_F^2 = \pm 2(A\pm XC)C^T \tag{8.18}$$

$$\nabla_X\|XC\|_F^2 = 2XCC^T \tag{8.19}$$

$$\nabla_X\|X\|_F^2 = 2X \tag{8.20}$$

$$\nabla_X\|A\circ X\|_F^2 = 2A\circ A\circ X \tag{8.21}$$

Alternate proof of (2):

$$\textbf{(2)} \quad \nabla_B \|D \circ (A \pm BC)\|_F^2 = \pm 2 \underbrace{(D \circ D \circ (A \pm BC))}_{m \times n} \underbrace{C^T}_{n \times p}$$

Proof. To show the identity we'll find the expression for an arbitrary element of this gradient.

$$\frac{\partial}{\partial B_{kl}} \|D \circ (A \pm BC)\|_F^2 = \frac{\partial}{\partial B_{kl}} \sum_i \sum_j \big(D_{ij}(A_{ij} \pm B_{i:}C_{:j})\big)^2 = \sum_i \sum_j \frac{\partial}{\partial B_{kl}} \big(D_{ij}(A_{ij} \pm B_{i:}C_{:j})\big)^2 =$$

$$\sum_j \frac{\partial}{\partial B_{kl}} \big(D_{kj}(A_{kj} \pm B_{k:}C_{:j})\big)^2 = 2 \sum_j D_{kj}(A_{kj} \pm B_{k:}C_{:j}) \frac{\partial}{\partial B_{kl}} D_{kj}(A_{kj} \pm B_{k:}C_{:j}) =$$

$$2 \sum_j D_{kj}^2 (A_{kj} \pm B_{k:}C_{:j}) \frac{\partial}{\partial B_{kl}} (A_{kj} \pm B_{k:}C_{:j}) = 2 \sum_j D_{kj}^2 (A_{kj} \pm B_{k:}C_{:j}) \Big(\frac{\partial}{\partial B_{kl}} A_{kj} \pm \frac{\partial}{\partial B_{kl}} B_{k:}C_{:j} \Big) =$$

$$2 \sum_j D_{kj}^2 (A_{kj} \pm B_{k:}C_{:j}) \Big(\pm \frac{\partial}{\partial B_{kl}} B_{k:}C_{:j} \Big) = \pm 2 \sum_j \Big(D_{kj}^2 (A_{kj} \pm B_{k:}C_{:j}) \frac{\partial}{\partial B_{kl}} \sum_p B_{kp}C_{pj} \Big) =$$

$$\pm 2 \sum_j D_{kj}^2 (A_{kj} \pm B_{k:}C_{:j}) \frac{\partial}{\partial B_{kl}} B_{kl}C_{lj} = \pm 2 \sum_j D_{kj}^2 (A_{kj} \pm B_{k:}C_{:j}) C_{lj} =$$

$$\pm 2 \Big((D \circ D)_{k:} \circ (A_{k:} \pm B_{k:}C) \Big) (C^\intercal)_{:l} = \pm 2 \Big(D \circ D \circ (A \pm BC)) C^T \Big)_{kl}$$

$$\Rightarrow \pm 2 (D \circ D \circ (A \pm BC)) C^T = \nabla_B \|D \circ (A \pm BC)\|_F^2$$

□

9 TF-IDF

- rare terms are not less relevant than frequent terms (IDF assumption);
- multiple occurrences of a term in a document are not less relevant than single occurrences (TF assumption);
- long documents are not preferred to short documents (normalization assumption).

In other words, terms that occur frequently in one document (TF =term-frequency), but rarely in the rest of the corpus (IDF = inverse-document-frequency), are more likely to be relevant to the topic of the document. In addition, normalizing the re- sulting weight vectors prevent longer documents from having a better chance of retrieval. [9, p. 78]

Let, t_k be a key term in the corpus, d_j a document in the corpus. Then let $f_{k,j}$ be the number of times a key term occurs in a document. Further, let Q_j be the set of key terms in a document. Then,

$$\mathrm{TF}(t_k, d_j) = \frac{f_{k,j}}{\max\limits_{z \in Q_j} f_{z,j}}$$

Let N be the number of documents in the corpus, and n_k be the number of documents in the corpus where t_k occurs at least once. Then,

$$\mathrm{IDF}(t_k, k_j) = \log \frac{N}{n_k}$$

So,

$$\text{TF-IDF}(t_k, d_j) = \mathrm{TF} * \mathrm{IDF} = \frac{f_{k,j}}{\max\limits_{z \in Q_j} f_{z,j}} \log \frac{N}{n_k}$$

TF-IDF weights, $w_{k,j}$, for term t_k, and document d_j are usually normalized to fall in the [0,1] interval using cosine normalization. Let t be the total number of key term types (as opposed to tokens) in the corpus.

$$w_{k,j} = \frac{\text{TF-IDF}(t_k, d_j)}{\sqrt{\sum_{s=1}^{t} \text{TF-IDF}(t_s, d_j)^2}}$$

[9, p. 78]

References

[1] M. Ekstrand, J. Riedl, and J. Konstan. Collaborative Filtering Recommendation Systems. Foundations and Trends in Human-Computer Interaction, Vol. 4, No. 2, pp. 81-173. 2011.

[2] L. Hu, J. Cao, G. Xu, L. Cao, Z. Gu, and C. Zhu. Personalized Recommendation via Cross-Domain Triadic Factorization. WWW 2013, May 13–17, 2013, Rio de Janiero, Brazil.

[3] Kolda, T.G. and Bader, B.W., 2009. Tensor decompositions and applications. SIAM review 51, 3, 455-500.

[4] Y. Koren and R. Bell. Advances in collaborative filtering. In Recommender Systems Handbook. Ricci et al. 2011, pages 145186.

[5] Y. Koren. Factorization meets the neighborhood: A multifaceted collaborative filtering model. In KDD, pages 426434, New York, NY, USA, 2008. ACM.

[6] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. IEEE Computer, 42(8):3037, 2009.

[7] J. Leskovec, J. McAuley. Hidden Factors and Hidden Topics: Understanding Rating Dimensions with Review Text. RecSys13, October 1216, 2013, Hong Kong, China.

[8] Petersen, Kaare Brandt, and Michael Syskind Pedersen. "The matrix cookbook." Technical University of Denmark 7 (2008): 15.

[9] F. Ricci et al. (eds.), Recommender Systems Handbook. New York: Springer, 2011

[10] Ricci, Giuseppe, Marco de Gemmis, and Giovanni Semeraro. "Matrix and Tensor Factorization Techniques applied to Recommender Systems: a Survey." Matrix 1.01 (2012).

[11] Saveski, Martin, and Amin Mantrach. "Item Cold-Start Recommendations: Learning Local Collective Embeddings". Proceedings of the 8th ACM Conference on Recommender systems. ACM, 2014.

[12] Karatzoglou, Alexandros, et al. "Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering." Proceedings of the fourth ACM conference on Recommender systems. ACM, 2010.

[13] Singh, Ajit P., and Geoffrey J. Gordon. "Relational learning via collective matrix factorization." Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2008.

[14] Takcs, Gbor, et al. "Matrix factorization and neighbor based algorithms for the netflix prize problem." Proceedings of the 2008 ACM conference on Recommender systems. ACM, 2008.