Fast Matrix Factorization for Online Recommendation with Implicit Feedback

Xiancong Chen (revised by Weike Pan)

College of Computer Science and Software Engineering
Shenzhen University

Reference: Fast Matrix Factorization for Online Recommendation with Implicit Feedback (SIGIR 2016) by Xiangnan He, Hanwang Zhang, Min-Yen Kan and Tat-Seng Chua.

Problem Definition

One-Class Collaborative Filtering (OCCF)

- Input: For each user $u \in \mathcal{U}$, we have a set of interacted items, i.e., $\mathcal{I}_u$.
- Goal: Recommend a ranked list of items (not associated with any interacted items) for each user u, i.e., $\mathcal{I}_u^{re} \subseteq \mathcal{I} \backslash \mathcal{I}_u$.

Notations

Table: Some notations and explanations.

n	user number
m	item number
$u \in \{1, 2, \ldots, n\}$	user ID
$i, k \in \{1, 2, \ldots, m\}$	item ID
$\mathcal{U}$	the whole set of users
$\mathcal{I}$	the whole set of items
$\mathcal{R} = \{(u, i)\}$	all (user, item) pairs
$\mathcal{I}_u$	the set of interacted items by user u
$\mathcal{U}_i$	the set of users who interact with item i
$U_{u\cdot} \in \mathbb{R}^{1\times d}$	user u's latent vector
$V_{i\cdot} \in \mathbb{R}^{1\times d}$	item i's latent vector
d	latent feature number
$\hat{r}_{ui}, \hat{r}_{uk}$	predictions of user u over interacted item i and uninteracted item k
ω_{ui}	weight of an interacted (user, item) pair (u, i)
s_k	item-oriented weight of item k in missing data
T	iteration number
λ	regularization parameter

Objective Function

$$
\begin{aligned}
L_{eALS} = &\sum_{u=1}^{n}\sum_{i\in\mathcal{I}_u}\omega_{ui}(\hat{r}_{ui}-r_{ui})^2+\sum_{u=1}^{n}\sum_{k\in\mathcal{I}\setminus\mathcal{I}_u}s_k\hat{r}_{uk}^2 \\
&+\lambda(\sum_{u=1}^{n}||U_{u\cdot}||^2+\sum_{i=1}^{m}||V_{i\cdot}||^2)
\end{aligned}
\tag{1}
$$

The first term denotes the prediction error of the interacted items of users. The second term is designed for the missing item k aiming to suppress $\hat{r}_{uk}$. Notice that ω_{ui} denotes the weight of an interacted (user, item) pair (u, i), and s_k denotes the confidence that item k un-interacted by users is a true negative assessment ($s_k = s_0\frac{f_k^\alpha}{\sum_{i=1}^{m} f_i^\alpha}$, where f_k denotes the popularity of item k).

Derivation (1/7)

We detail the derivation process for U_{uf}, and the process for V_{if} is achieved likewise. For convenience, we set $L_1 = \sum_{u=1}^{n} \sum_{i \in \mathcal{I}_u} \omega_{ui} (\hat{r}_{ui} - r_{ui})^2, L_2 = \sum_{u=1}^{n} \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} s_k \hat{r}_{uk}^2, L_{Reg} = \lambda (\sum_{u=1}^{n} ||U_{u\cdot}||^2 + \sum_{i=1}^{m} ||V_{i\cdot}||^2)$. Then, we get the derivation of the objective function w.r.t. U_{uf}:

$$\frac{\partial L_{eALS}}{\partial U_{uf}} = \frac{\partial (L_1 + L_2 + L_{Reg})}{\partial U_{uf}} \tag{2}$$

Notice that the parameters are optimized at the element level. The prediction without the component of latent factor f is as follows:

$$\hat{r}_{ui}^{-f} = \hat{r}_{ui} - U_{uf} V_{if} \tag{3}$$

Derivation (2/7)

The derivation of $\frac{\partial L_1}{\partial U_{uf}}$ is as follows:

$$\begin{aligned}\frac{\partial L_1}{\partial U_{uf}} &= \frac{\partial \sum\limits_{i\in\mathcal{I}_u} \omega_{ui}(\hat{r}_{ui} - r_{ui})^2}{\partial U_{uf}} \\ &= \sum_{i\in\mathcal{I}_u} 2\omega_{ui}(\hat{r}_{ui} - r_{ui})V_{if} \\ &= \sum_{i\in\mathcal{I}_u} 2\omega_{ui}(\hat{r}_{ui}^{-f} + U_{uf}V_{if} - r_{ui})V_{if} \\ &= \sum_{i\in\mathcal{I}_u} 2\omega_{ui}(\hat{r}_{ui}^{-f} - r_{ui})V_{if} + \sum_{i\in\mathcal{I}_u} 2\omega_{ui}U_{uf}V_{if}^2\end{aligned} \tag{4}$$

Derivation (3/7)

The derivation of $\frac{\partial L_2}{\partial U_{uf}}$ is as follows:

$$
\begin{aligned}
\frac{\partial L_2}{\partial U_{uf}} &= \frac{\partial \sum\limits_{k \in \mathcal{I} \setminus \mathcal{I}_u} s_k \hat{r}_{uk}^2}{\partial U_{uf}} \\
&= \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} 2 s_k \hat{r}_{uk} V_{kf} \\
&= \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} 2 s_k (\hat{r}_{uk}^{-f} + U_{uf} V_{kf}) V_{kf} \\
&= \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} 2 s_k \hat{r}_{uk}^{-f} V_{kf} + \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} 2 s_k U_{uf} V_{kf}^2
\end{aligned}
\tag{5}
$$

Derivation (4/7)

The derivation of $\frac{\partial L_{Reg}}{\partial U_{uf}}$ is as follows:

$$\frac{\partial L_{reg}}{\partial U_{uf}} = 2\lambda U_{uf} \tag{6}$$

According to Eqs.(4 - 6), we can get the derivation of $\frac{\partial L_{eALS}}{\partial U_{uf}}$:

$$\begin{aligned}\frac{\partial L_{eALS}}{\partial U_{uf}} &= \sum_{i \in \mathcal{I}_u} 2\omega_{ui}(\hat{r}_{ui}^{-f} - r_{ui})V_{if} + \sum_{i \in \mathcal{I}_u} 2\omega_{ui}U_{uf}V_{if}^2 \\ &+ \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} 2s_k \hat{r}_{uk}^{-f} V_{kf} + \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} 2s_k U_{uf} V_{kf}^2 + 2\lambda U_{uf}\end{aligned} \tag{7}$$

Derivation (5/7)

By setting $\frac{\partial L_{eALS}}{\partial U_{uf}} = 0$, we obtain the update rule of U_{uf}:

$$U_{uf} = \frac{-\sum_{i \in \mathcal{I}_u} \omega_{ui}(\hat{r}_{ui}^{-f} - r_{ui})V_{if} - \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} s_k \hat{r}_{uk}^{-f} V_{kf}}{\sum_{i \in \mathcal{I}_u} \omega_{ui} V_{if}^2 + \sum_{k \in \mathcal{I} \setminus \mathcal{I}_u} s_k V_{kf}^2 + \lambda} \tag{8}$$

The computational bottleneck lies in the summation over the missing data portion. To solve this problem, the memoization strategy can be applied.

Derivation (6/7)

We define the $\mathbf{C}^q$ cache as $\mathbf{C}^q = \sum_{k=1}^{m} s_k V_{k\cdot}^T V_{k\cdot} \in \mathbb{R}^{d\times d}$, which can be pre-computed. Then we can get:

$$
\begin{aligned}
\sum_{k\in\mathcal{I}\setminus\mathcal{I}_u} s_k \hat{r}_{uk}^{-f} V_{kf} &= \sum_{k=1}^{m} s_k V_{kf} \sum_{e\neq f} U_{ue} V_{ke} - \sum_{k\in\mathcal{I}_u} s_k \hat{r}_{uk}^{-f} V_{kf} \\
&= \sum_{e\neq f} U_{ue} \sum_{k=1}^{m} s_k V_{kf} V_{ke} - \sum_{k\in\mathcal{I}_u} s_k \hat{r}_{uk}^{-f} V_{kf} \qquad (9) \\
&= \sum_{e\neq f} U_{ue} c_{fe}^{q} - \sum_{k\in\mathcal{I}_u} s_k \hat{r}_{uk}^{-f} V_{kf}
\end{aligned}
$$

$$
\sum_{k\in\mathcal{I}\setminus\mathcal{I}_u} s_k V_{kf}^2 = \sum_{k=1}^{m} s_k V_{kf}^2 - \sum_{k\in\mathcal{I}_u} s_k V_{kf}^2 = c_{ff}^{q} - \sum_{k\in\mathcal{I}_u} s_k V_{kf}^2 \qquad (10)
$$

Derivation (7/7)

By leveraging the above memoization strategy, the update rule for U_{uf} with the use of $\mathbf{C}^q$ cache is as follows:

$$U_{uf} = \frac{\sum_{i \in \mathcal{I}_u} [\omega_{ui} r_{ui} - (\omega_{ui} - s_i)\hat{r}_{ui}^{-f}] V_{if} - \sum_{e \neq f} U_{ue} c_{ef}^q}{\sum_{i \in \mathcal{I}_u} (\omega_{ui} - s_i) V_{if}^2 + c_{ff}^q + \lambda} \tag{11}$$

which can be done in $O(d + |\mathcal{I}_u|)$ time.
Similarly, we can derive the update rule for V_{if}:

$$V_{if} = \frac{\sum_{u \in \mathcal{U}_i} [\omega_{ui} r_{ui} - (\omega_{ui} - s_i)\hat{r}_{ui}^{-f}] U_{uf} - s_i \sum_{e \neq f} V_{ie} c_{ef}^p}{\sum_{u \in \mathcal{U}_i} (\omega_{ui} - s_i) U_{uf}^2 + s_i c_{ff}^p + \lambda} \tag{12}$$

which can be done in $O(d + |\mathcal{U}_i|)$ time. Notice that c_{ef}^p denotes the $(e, f)^{th}$ element of the $\mathbf{C}^p$ cache, defined as $\mathbf{C}^p = \sum_{k=1}^{n} U_{k\cdot}^T U_{k\cdot} \in \mathbb{R}^{d \times d}$.

Input: $\mathcal{R}, d, \lambda, \omega_{ui}, s_k$;
Output: Latent feature vectors $U_{u\cdot}$ and $V_{i\cdot}$;
Randomly initialize $U_{u\cdot}$ and $V_{i\cdot}$;
for $(u, i) \in \mathcal{R}$ **do** $\hat{r}_{ui} = U_{u\cdot} V_{i\cdot}^T$;
while *Stopping criteria is not met* **do**

$$\mathbf{C}^q = \sum_{k=1}^{m} s_k V_{k\cdot}^T V_{k\cdot};$$

  for $u = 1, \ldots, n$ **do**
    for $f = 1, \ldots, d$ **do**
      for $i = 1, \ldots, |I_u|$ **do**
        $\hat{r}_{ui}^{-f} = \hat{r}_{ui} - U_{uf} V_{if}$;
      end for
      Update U_{uf} according to Eq.(11);
      for $i = 1, \ldots, |I_u|$ **do**
        $\hat{r}_{ui} = \hat{r}_{ui}^{-f} + U_{uf} V_{if}$;
      end for
    end for
  end for
  $\mathbf{C}^p = \sum_{k=1}^{n} U_{k\cdot}^T U_{k\cdot}$;
  for $i = 1, \ldots, m$ **do**
    for $f = 1, \ldots, d$ **do**
      for $u = 1, \ldots, |U_i|$ **do**
        $\hat{r}_{ui}^{-f} = \hat{r}_{ui} - U_{uf} V_{if}$;
      end for
      Update V_{if} according to Eq.(12);
      for $u = 1, \ldots, |U_i|$ **do**
        $\hat{r}_{ui} = \hat{r}_{ui}^{-f} + U_{uf} V_{if}$;
      end for
    end for
  end for
end while

Time Complexity

- ALS [Hu et al., 2008]: $O((m+n)d^3 + |\mathcal{R}|d^2)$
- BPR [Rendle et al., 2009]: $O(|\mathcal{R}|d)$
- eALS: $O((m+n)d^2 + |\mathcal{R}|d)$

Datasets and Evaluation Metrics

Table: Description of the dataset used in the experiments.

Dataset	$\lvert\mathcal{U}\rvert$	$\lvert\mathcal{I}\rvert$	$\lvert\mathcal{R}^{\mathcal{P}}\rvert$	$\lvert\mathcal{R}^{\mathcal{P}^{val}}\rvert$	$\lvert\mathcal{R}^{\mathcal{P}^{te}}\rvert$
Netflix	480,189	17,770	4,554,888	4,556,347	4,558,508

- We take the following steps for data preprocessing: (i) we first randomly take 60% (user, item, rating) triples and keep the (user, item) pairs with rating value 5 as purchases; (ii) we then divide them into three parts with equal size, one part for training, one part for validation, and the left part for test. We repeat this procedure for three times in order to obtain three copies of data.
- For performance evaluation, we use five commonly used ranking-oriented metrics, including precision@5, recall@5, f1@5, ndcg@5 and 1-call@5.

Baselines

Baseline

- Bayesian Personalized Ranking (BPR) [Rendle et al., 2009]

Parameter Configurations

- For BPP, we fix the number of latent dimensions $d = 20$ and the learning rate $\gamma = 0.01$, and search the best value of the tradeoff parameters from $\{0.001, 0.01, 0.1\}$ and the best iteration number T from $\{100, 500, 1000\}$ according to the performance of NDCG@15 on the validation data.
- For eALS , we fix the number of latent dimensions $d = 20$, and search the best value of λ from $\{0.001, 0.01, 0.1\}$, the best value of $s_0 (s_k = s_0/m)$ from $\{100, 200, 400, 800, 1600, 3200, 6400\}$ and the best iteration number T from $\{50, 100, 200\}$ according to the performance of NDCG@15 on the validation data.

Results

Table: Recommendation performance on Netflix.

Dataset	Method	Precision@5	Recall@5	F1@5	NDCG@5	1-call@5
Netflix	BPR	0.0716± 0.0007	0.0480± 0.0005	0.0446± 0.0005	0.0818± 0.0011	0.2846± 0.0022
	eALS	0.0806± 0.0003	0.0559± 0.0003	0.0519± 0.0002	0.0931± 0.0004	0.3150± 0.0012

Conclusion

The recommendation method eALS is efficient and effective with the memoization strategy.

He, X., Zhang, H., Kan, M.-Y., and Chua, T.-S. (2016).
Fast matrix factorization for online recommendation with implicit feedback.
In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'16, pages 549–558.

Hu, Y., Koren, Y., and Volinsky, C. (2008).
Collaborative filtering for implicit feedback datasets.
In *Proceedings of the 8th IEEE International Conference on Data Mining*, ICDM'087, pages 263–272.

Rendle, S., Freudenthaler, C., Gantner, Z., and Schmidt-Thieme, L. (2009).
BPR: Bayesian personalized ranking from implicit feedback.
In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, UAI'09, pages 452–461.