Transfer Learning for Semi-Supervised Collaborative Recommendation

Weike Pan¹, Qiang Yang²*, Yuchao Duan¹ and Zhong Ming¹*
panweike@szu.edu.cn, qyang@cse.ust.hk, duanyuchao@email.szu.edu.cn, mingz@szu.edu.cn

¹College of Computer Science and Software Engineering
Shenzhen University, Shenzhen, China

²Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong, China
Problem Definition

Semi-Supervised Collaborative Recommendation (SSCR)

- **Input:**
  - **Labeled feedback** (or explicit feedback) $\mathcal{R} = \{(u, i, r_{ui})\}$: the rating $r_{ui}$ and the corresponding (user, item) pair $(u, i)$ is a kind of a **real-valued label** and a featureless instance, respectively.
  - **Unlabeled feedback** (or implicit feedback) $\mathcal{O} = \{(u, i)\}$: the (user, item) pair $(u, i)$ is an **unlabeled instance** without supervised information.

- **Goal:** predict the preference of each (user, item) pair in the test data $\mathcal{R}^{te}$. 
Challenges

- The **heterogeneity** challenge: how to integrate two different types of feedback (explicit and accurate preferences vs. implicit and uncertain preferences).
- The **uncertainty** challenge: how to identify some likely-positive feedback from the unlabeled feedback associated with high uncertainty w.r.t. users’ true preferences.
Overall of Our Solution

We map the SSCR problem to the transfer learning paradigm, and design an iterative algorithm, *Self-Transfer Learning* (sTL), containing two basic steps:

1. For the first step of knowledge flow from the unlabeled feedback to the labeled feedback, we focus on integrating the identified likely-positive unlabeled feedback into the learning task of labeled feedback.

2. For the second step of knowledge flow from the labeled feedback to the unlabeled feedback, we turn to use the tentatively learned model for further identification of likely-positive unlabeled feedback.
Advantages of Our Solution

The unlabeled-to-labeled knowledge flow and labeled-to-unlabeled knowledge flow can address the heterogeneity challenge and the uncertainty challenge, respectively.

The iterative algorithm is able to achieve sufficient knowledge transfer between labeled feedback and unlabeled feedback.
Table: Some notations (part 1).

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>user number</td>
</tr>
<tr>
<td>$m$</td>
<td>item number</td>
</tr>
<tr>
<td>$u$</td>
<td>user ID</td>
</tr>
<tr>
<td>$i, i'$</td>
<td>item ID</td>
</tr>
<tr>
<td>$r_{ui}$</td>
<td>observed rating of $(u, i)$</td>
</tr>
<tr>
<td>$\hat{r}_{ui}$</td>
<td>predicted rating of $(u, i)$</td>
</tr>
<tr>
<td>$\mathcal{R} = {(u, i, r_{ui})}$</td>
<td>labeled feedback (training)</td>
</tr>
<tr>
<td>$\mathcal{O} = {(u, i)}$</td>
<td>unlabeled feedback (training)</td>
</tr>
<tr>
<td>$\mathcal{R}<em>{te} = {(u, i, r</em>{ui})}$</td>
<td>labeled feedback (test)</td>
</tr>
<tr>
<td>$\tilde{I}_u = {i}$</td>
<td>examined items by user $u$</td>
</tr>
</tbody>
</table>
**Table:** Some notations (part 2).

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mu \in \mathbb{R}$</td>
<td>global average rating value</td>
</tr>
<tr>
<td>$b_u \in \mathbb{R}$</td>
<td>user bias</td>
</tr>
<tr>
<td>$b_i \in \mathbb{R}$</td>
<td>item bias</td>
</tr>
<tr>
<td>$d \in \mathbb{R}$</td>
<td>number of latent dimensions</td>
</tr>
<tr>
<td>$U_{u.} \in \mathbb{R}^{1 \times d}$</td>
<td>user-specific feature vector</td>
</tr>
<tr>
<td>$U \in \mathbb{R}^{n \times d}$</td>
<td>user-specific feature matrix</td>
</tr>
<tr>
<td>$V_{i.}, W^{(s)}_{i.} \in \mathbb{R}^{1 \times d}$</td>
<td>item-specific feature vector</td>
</tr>
<tr>
<td>$V, W^{(s)} \in \mathbb{R}^{m \times d}$</td>
<td>item-specific feature matrix</td>
</tr>
<tr>
<td>$T, L$</td>
<td>iteration number</td>
</tr>
</tbody>
</table>
Prediction Rule of sTL

The predicted preference of user $u$ on item $i$,\[ \hat{r}_{ui}^{(\ell)} = \mu + b_u + b_i + U_u \cdot V_i^T + \sum_{s=0}^{\ell} \bar{U}_u(s) V_i^T, \] (1)

where $\bar{U}_u(s) = \frac{1}{\sqrt{|\tilde{I}_u|}} \sum_{i' \in \tilde{I}_u(s)} W_i(s), \tilde{I}_u(0) = \tilde{I}_u$ and $\tilde{I}_u(s) \subseteq \tilde{I}_u$.

Note that when $\ell = 0$, the above prediction rule is exactly the same with that of SVD++. 
Objective Function of sTL

The optimization problem,

\[
\min_{\mathcal{I}(\ell), \Theta(\ell)} \sum_{u=1}^{n} \sum_{i=1}^{m} y_{ui} \left[ \frac{1}{2} (r_{ui} - \hat{r}_{ui}^{(\ell)})^2 + \text{reg}(\Theta(\ell)) \right],
\]

where \(\mathcal{I}(\ell) = \{\tilde{\mathcal{I}}_u^{(s)}\}_{s=0}^{\ell}\) and \(\Theta(\ell) = \{\mu, b_u, b_i, U_u, V_i, W_i^{(s)}\}_{s=0}^{\ell}\) are likely-to-prefer items to be identified and model parameters to be learned, respectively. The regularization term

\[
\text{reg}(\Theta(\ell)) = \frac{\lambda}{2} \|U_u\|^2 + \frac{\lambda}{2} \|V_i\|^2 + \frac{\lambda}{2} \|b_u\|^2 + \frac{\lambda}{2} \|b_i\|^2 + \\
\frac{\lambda}{2} \sum_{s=0}^{\ell} \sum_{i' \in \mathcal{I}_u^{(s)}} \|W_i^{(s)}\|^2 + \frac{\lambda}{2} \sum_{s=1}^{\ell} \sum_{i' \in \mathcal{I}_u^{(s)}} \|W_i^{(s)} - W_i^{(0)}\|^2
\]

is used to avoid overfitting. In particular, the term

\[
\sum_{s=1}^{\ell} \sum_{i' \in \mathcal{I}_u^{(s)}} \|W_i^{(s)} - W_i^{(0)}\|^2
\]

will constrain \(W_i^{(s)}\) to be similar to \(W_i^{(0)}\), which is helpful to avoid overfitting when \(W_i^{(s)}\) is associated with insufficient training data, i.e., when \(|\mathcal{I}_u^{(s)}|\) is small.
Learning the sTL (1/3)

For the first step of unlabeled-to-labeled knowledge flow, we adopt a gradient descent algorithm to learn the model parameters. We denote \( g_{ui} = \frac{1}{2}(r_{ui} - \hat{r}_{ui}^{(\ell)})^2 + \text{reg}(\Theta^{(\ell)}) \) and have the gradient,

\[
\frac{\partial g_{ui}}{\partial \theta} = (r_{ui} - \hat{r}_{ui}^{(\ell)}) \frac{\partial \hat{r}_{ui}^{(\ell)}}{\partial \theta} + \frac{\partial \text{reg}(\Theta^{(\ell)})}{\partial \theta},
\]

where \( \theta \) can be \( \mu, b_u, b_i, U_u ., V_i . \) and \( W_{i'}^{(s)} \), and the gradient thus includes

\[
\begin{align*}
\frac{\partial g_{ui}}{\partial \mu} &= -e_{ui}, \\
\frac{\partial g_{ui}}{\partial b_u} &= -e_{ui} + \lambda b_u, \\
\frac{\partial g_{ui}}{\partial b_i} &= -e_{ui} + \lambda b_i, \\
\frac{\partial g_{ui}}{\partial U_u .} &= -e_{ui} V_i . + \lambda U_u ., \\
\frac{\partial g_{ui}}{\partial V_i .} &= -e_{ui} (U_u . + \sum_{s=0}^{\ell} \tilde{U}_{u .}^{(s)}) + \lambda V_i ., \text{ and} \\
\frac{\partial g_{ui}}{\partial W_{i'}^{(s)}} &= -e_{ui} \frac{1}{\sqrt{|\tilde{I}_{u}^{(s)}|}} V_i . + \lambda W_{i'}^{(s)} + \lambda(W_{i'}^{(s)} - W_{i'}^{(0)}) \text{ with} \\
i' &\in \tilde{I}_{u}^{(s)}, s = 0, \ldots, \ell. \text{ Note that } e_{ui} = r_{ui} - \hat{r}_{ui}^{(\ell)} \text{ denotes the difference between the true rating and predicted rating.} 
\end{align*}
\]
Learning the sTL (2/3)

We then have the update rule for each model parameter,

$$\theta = \theta - \gamma \frac{\partial g_{ui}}{\partial \theta},$$  \hspace{1cm} (4)

where $\theta$ again can be $\mu$, $b_u$, $b_i$, $U_u$, $V_i$ and $W^{(s)}_{ji}$, and $\gamma$ ($\gamma > 0$) is the step size or learning rate when updating the model parameters.
For the second step of labeled-to-unlabeled knowledge flow, we use the latest learned model parameters and the accumulated identified items, i.e., $\mathcal{I}^{(s)}$ and $\Theta^{(s)}$, to construct $\tilde{\mathcal{I}}^{(s+1)}_u$ for each user $u$:

- we estimate the preference of user $u$ on item $i$ for each $(u, i) \in \mathcal{O}$, i.e., $\hat{r}_{ui}^{(s)}$, via the prediction rule in Eq.(1)
- we remove the (user, item) pair $(u, i)$ from $\mathcal{O}$ and put the item $i$ in $\tilde{\mathcal{I}}^{(s+1)}_u$ if $\hat{r}_{ui}^{(s)} > r_0$, where $r_0$ is a predefined threshold

Note that with the newly identified item set $\tilde{\mathcal{I}}^{(s+1)}_u$, we can integrate them into the learning task of labeled feedback again.
Algorithm (1/2)

1: **Input**: Labeled and unlabeled feedback $R$, $O$; tradeoff parameter $\lambda$, threshold $r_0$, latent dimension number $d$, and iteration numbers $L$, $T$.

2: **Output**: Learned model parameters $\Theta^{(L)}$ and identified likely-to-prefer items $I_u^{(s)}$, $s = 1, \ldots, L$.

3: **Initialization**: Initialize the item set $I_u^{(0)} = \tilde{I}_u$ for each user $u$.

4: **for** $\ell = 0, \ldots, L$ **do**

5: **Please see the details in the next page**

6: **end for**
1: // Step 1: Unlabeled-to-labeled knowledge flow
2: Set the learning rating $\gamma = 0.01$ and initialize the model parameters $\Theta^{(\ell)}$
3: for $t = 1, \ldots, T$ do
4:   for $t_2 = 1, \ldots, |R|$ do
5:     Randomly pick up a rating record $(u, i, r_{ui})$ from $R$
6:     Calculate the gradients $\frac{\partial g_{ui}}{\partial \theta}$
7:     Update the model parameters $\theta$
8:   end for
9:   Decrease the learning rate $\gamma \leftarrow \gamma \times 0.9$
10: end for
11: // Step 2: Labeled-to-unlabeled knowledge flow
12: if $\ell < L$ then
13:   for $u = 1, \ldots, n$ do
14:     Predict the preference $\hat{r}_{ui}', i' \in \tilde{I}_u \setminus \bigcup_{s=1}^{\ell} \tilde{I}_u^{(s)}$
15:     Select some likely-to-prefer items from $\tilde{I}_u \setminus \bigcup_{s=1}^{\ell} \tilde{I}_u^{(s)}$ with $\hat{r}_{ui} > r_0$ and save them as $\tilde{I}_u^{(\ell+1)}$
16:   end for
17: end if
Analysis

The whole algorithm iterates in $L + 1$ loops:

- When $L = 0$, the sTL algorithm reduces to a single step of unlabeled-to-labeled knowledge flow, which is the same with that of SVD++ using the whole unlabeled feedback without uncertainty reduction.
- When $L = 0$ and $\mathcal{O} = \emptyset$, sTL further reduces to the basic matrix factorization.

We illustrate the relationships among sTL, SVD++ and MF as follows,

$$
\text{sTL} \xrightarrow{L=0} \text{SVD++} \xrightarrow{\mathcal{O}=\emptyset} \text{MF},
$$

(5)

from which we can see that our sTL is a quite generic algorithm.
## Datasets

**Table:** Statistics of one copy of labeled feedback $\mathcal{R}$, unlabeled feedback $\mathcal{O}$ and test records $\mathcal{R}^{te}$ of ML10M, Flixter and ML20M used in the experiments.

<table>
<thead>
<tr>
<th></th>
<th>ML10M</th>
<th>Flixter</th>
<th>ML20M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labeled feedback</td>
<td>$(u, i, r_{ui})$, $r_{ui} \in {0.5, 1, \ldots, 5}$</td>
<td>$(u, i)$</td>
<td>$(u, i, r_{ui})$, $r_{ui} \in {0.5, 1, \ldots, 5}$</td>
</tr>
<tr>
<td>Unlabeled feedback</td>
<td>$(u, i)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test feedback</td>
<td>$(u, i, r_{ui})$, $r_{ui} \in {0.5, 1, \ldots, 5}$</td>
<td>$(u, i)$</td>
<td>$(u, i, r_{ui})$, $r_{ui} \in {0.5, 1, \ldots, 5}$</td>
</tr>
<tr>
<td>User # ($n$)</td>
<td>71567</td>
<td>147612</td>
<td>138493</td>
</tr>
<tr>
<td>Item # ($m$)</td>
<td>10681</td>
<td>48794</td>
<td>26744</td>
</tr>
<tr>
<td>Labeled feedback # ($</td>
<td>\mathcal{R}</td>
<td>$)</td>
<td>4000022</td>
</tr>
<tr>
<td>Unlabeled feedback # ($</td>
<td>\mathcal{O}</td>
<td>$)</td>
<td>4000022</td>
</tr>
<tr>
<td>Test feedback # ($</td>
<td>\mathcal{R}^{te}</td>
<td>$)</td>
<td>2000010</td>
</tr>
</tbody>
</table>
Experiments

Baselines

- Item-based collaborative filtering (ICF)
- Matrix factorization (MF)
- SVD with unlabeled feedback (SVD++)
- Factorization machine (FM)
Initialization of Model Parameters

We use the statistics of the training data $R$ to initialize the model parameters:

- For each entry of matrix $U$, $V$ and $W^{(s)}$, i.e., $U_{uk}$, $V_{ik}$ and $W^{(s)}_{i'k}$ with $k = 1, \ldots, d$ and $s = 1, \ldots, \ell$, we use a small random value $(r - 0.5) \times 0.01$, where $r \in [0, 1)$ is a small random number.

- For the bias of user $u$ and item $i$, i.e., $b_u$ and $b_i$, we use $b_u = \bar{r}_u - \mu$ and $b_i = \bar{r}_i - \mu$, where $\bar{r}_u$, $\bar{r}_i$, $\mu$ are user $u$’s average rating, item $i$’s average rating and global rating, respectively.
Experiments

Parameter Configurations

- For the number of latent dimensions $d$ and the iteration number $T$, we set them as $d = 20$ and $T = 100$.

- For the tradeoff parameter $\lambda$, we search it from $\lambda \in \{0.001, 0.01, 0.1\}$ using RMSE on the first copy of each data (via sampling a holdout validation data with $n$ records from the training data) and then fix it for the remaining two copies.

- For the threshold $r_0$, we first set it close to the average rating of each data set, i.e., $r_0 = 3.5$, and then study the impact of using smaller and bigger values.

- For the number of knowledge transfer steps $L$ in our sTL, we first fix it as $L = 2$, and then study the performance with different values of $L \in \{0, 1, 2, 3, 4\}$.

- We set the number of neighbors as 50 in ICF.
When we estimate the preference for user $u$ on item $i$, i.e., $\hat{r}_{ui}$, the predicted rating may be out of the range of labeled feedback of the training data, i.e., $[0.5, 5]$ for the data sets in our experiments. For a predicted preference that is larger than 5 or smaller than 0.5, we adopt the following commonly used post-processing before final evaluation,

$$
\hat{r}_{ui} = \begin{cases} 
0.5, & \text{if } \hat{r}_{ui} < 0.5 \\
5, & \text{if } \hat{r}_{ui} > 5
\end{cases}
$$

(6)
Evaluation Metrics

- **Mean Absolute Error (MAE)**

\[
MAE = \sum_{(u,i,r_{ui}) \in \mathcal{R}^{te}} \frac{|r_{ui} - \hat{r}_{ui}|}{|\mathcal{R}^{te}|}
\]

- **Root Mean Square Error (RMSE)**

\[
RMSE = \sqrt{\sum_{(u,i,r_{ui}) \in \mathcal{R}^{te}} \frac{(r_{ui} - \hat{r}_{ui})^2}{|\mathcal{R}^{te}|}}
\]
Table: The significantly best results are marked in bold font (the p-values are smaller than 0.01).

<table>
<thead>
<tr>
<th>Data</th>
<th>Method</th>
<th>MAE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML10M (R)</td>
<td>ICF</td>
<td>0.6699±0.0003</td>
<td>0.8715±0.0004</td>
</tr>
<tr>
<td></td>
<td>MF</td>
<td>0.6385±0.0008</td>
<td>0.8323±0.0011</td>
</tr>
<tr>
<td>ML10M (R, O)</td>
<td>SVD++</td>
<td>0.6249±0.0006</td>
<td>0.8182±0.0009</td>
</tr>
<tr>
<td></td>
<td>FM</td>
<td>0.6276±0.0004</td>
<td>0.8181±0.0006</td>
</tr>
<tr>
<td></td>
<td>sTL</td>
<td>0.6209±0.0004</td>
<td>0.8103±0.0007</td>
</tr>
<tr>
<td>Flixter (R)</td>
<td>ICF</td>
<td>0.6687±0.0007</td>
<td>0.9061±0.0010</td>
</tr>
<tr>
<td></td>
<td>MF</td>
<td>0.6479±0.0007</td>
<td>0.8749±0.0010</td>
</tr>
<tr>
<td>Flixter (R, O)</td>
<td>SVD++</td>
<td>0.6400±0.0008</td>
<td>0.8683±0.0009</td>
</tr>
<tr>
<td></td>
<td>FM</td>
<td>0.6447±0.0007</td>
<td>0.8701±0.0008</td>
</tr>
<tr>
<td></td>
<td>sTL</td>
<td>0.6398±0.0006</td>
<td>0.8650±0.0008</td>
</tr>
<tr>
<td>ML20M (R)</td>
<td>ICF</td>
<td>0.6555±0.0002</td>
<td>0.8591±0.0004</td>
</tr>
<tr>
<td></td>
<td>MF</td>
<td>0.6226±0.0005</td>
<td>0.8153±0.0007</td>
</tr>
<tr>
<td>ML20M (R, O)</td>
<td>SVD++</td>
<td>0.6122±0.0004</td>
<td>0.8033±0.0006</td>
</tr>
<tr>
<td></td>
<td>FM</td>
<td>0.6120±0.0004</td>
<td>0.8036±0.0007</td>
</tr>
<tr>
<td></td>
<td>sTL</td>
<td>0.6064±0.0002</td>
<td>0.7969±0.0004</td>
</tr>
</tbody>
</table>
Observations

- The proposed self-transfer learning (sTL) algorithm achieves better performance than all other baselines in all cases. Such significant superiority in preference prediction clearly shows the advantage of the designed knowledge flow strategy in sTL in order to fully leverage the uncertain unlabeled feedback in an iterative manner.

- The overall ordering w.r.t. preference prediction performance is $\text{ICF} < \text{MF} < \text{SVD++} < \text{FM} < \text{sTL}$, which shows that (i) the uncertain unlabeled feedback are useful for preference learning, and (ii) SVD++ and FM are indeed very strong baselines for digesting labeled and unlabeled feedback in a principled way.
Table: Summary of some related works on collaborative recommendation, including supervised, unsupervised and semi-supervised collaborative recommendation settings for labeled feedback $R$, unlabeled feedback $O$, and heterogeneous feedback $R$ and $O$, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised ($R$)</td>
<td>ICF, etc.: memory-based method</td>
</tr>
<tr>
<td></td>
<td>MF, etc.: model-based method</td>
</tr>
<tr>
<td>Unsupervised ($O$)</td>
<td>IMF, etc.: with pointwise assumption</td>
</tr>
<tr>
<td></td>
<td>BPR, etc.: with pairwise assumption</td>
</tr>
<tr>
<td>Semi-Supervised ($R, O$)</td>
<td>SVD++, FM, etc.: for heterogeneity challenge</td>
</tr>
<tr>
<td></td>
<td>$sTL$ (proposed): for heterogeneity &amp; uncertainty challenges</td>
</tr>
</tbody>
</table>
Transfer Learning for Collaborative Recommendation

Most previous works on transfer learning for collaborative recommendation are somehow one-time knowledge transfer, i.e., the algorithm only contains a step of unlabeled-to-labeled knowledge flow represented by one single arrowed line from right to left.

We generalize the commonly adopted one-time knowledge transfer approach in previous works, and design a novel iterative knowledge transfer algorithm, i.e., self-transfer learning, aiming to address the heterogeneity and uncertainty challenges of the labeled and unlabeled feedback in one single framework.
We study an important problem with both labeled feedback (explicit feedback) and unlabeled feedback (implicit feedback), i.e., semi-supervised collaborative recommendation (SSCR), in the transfer learning paradigm.

We design a novel transfer learning algorithm, i.e., self-transfer learning (sTL), which is able to identify and integrate likely-positive unlabeled feedback into the learning task of labeled feedback in a principled and iterative manner.
Thank you!

- We thank the editors and reviewers for their expert comments and constructive suggestions.
- Weike Pan, Yuchao Duan and Zhong Ming thank the support of National Natural Science Foundation of China (NSFC) Nos. 61502307 and 61170077, Natural Science Foundation of Guangdong Province Nos. 2014A030310268 and 2016A030313038, and Natural Science Foundation of SZU No. 201436.
- Qiang Yang thanks the support of China National 973 project 2014CB340304, and Hong Kong CERG projects 16211214 and 16209715.