Discrete Collaborative Filtering

Hanwang Zhang\textsuperscript{1}, Fumin Shen\textsuperscript{2}, Wei Liu\textsuperscript{3},
\textbf{Xiangnan He}\textsuperscript{1}, Huanbo Luan\textsuperscript{4}, Tat-Seng Chua\textsuperscript{1}

Presented by Xiangnan He

1. National University of Singapore
2. University of Electronic Science and Technology of China
3. Tencent Research
4. Tsinghua University

\textsuperscript{19} July 2016
Online Recommendation
• An *Efficient* Recommender System
• Latent Model: *Binary* Representation for Users and Items
• Recommendation as *Search* with Binary Codes

Offline Training
• *End-to-end* binary optimization
• *Balanced* and *Decorrelated* Constraint
• Small *SVD + Discrete* Coordinate Descent
Collaborative Filtering

Efficient CF: Hashing Users & Items

Recommendation is **Search**

- Ranking by `<user vector, item vector>`

Search in Euclidean space is **slow**

- Requires float operations & linear scan of the data

Search in **Hamming Space** is **fast**.

- Only requires XOR operation & constant-time lookup

---

User-Item Database

<table>
<thead>
<tr>
<th>Query Code</th>
<th>Hash Table</th>
<th>User-Item Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1010</td>
<td>010010</td>
</tr>
<tr>
<td>0</td>
<td>1000</td>
<td>10011</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>1</td>
<td>1100</td>
<td>00101</td>
</tr>
</tbody>
</table>
Stage 1: Relaxed Real-Valued Problem

\{B, D\} \leftarrow \text{Continuous CF Methods}

Stage 2: Binarization

\begin{align*}
B &\leftarrow \text{sgn} \ (B), \\
D &\leftarrow \text{sgn} \ (D)
\end{align*}

Code learning and CF are isolated

Quantization Loss
1. A,B,a,b are close but they are separated into different quadrants
2. C, d should be far but they are assigned to the same quadrant
## Tackling Quantization Loss

### User-Item Matrix

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>.8</td>
<td>.4</td>
<td>?</td>
</tr>
<tr>
<td>B</td>
<td>.8</td>
<td>.8</td>
<td>?</td>
<td>.4</td>
</tr>
<tr>
<td>C</td>
<td>.2</td>
<td>?</td>
<td>.2</td>
<td>.4</td>
</tr>
</tbody>
</table>

---

**Relaxed Solution**

- $A \cdot a^c$
- $B \cdot b^d$
- $C$

**Round-off**

- $-1 +1$
- $+1 +1$
- $-1 -1$
- $+1 -1$

**Discrete Optimization**

- $AB \cdot a \ b$
- $c$
- $d$
- $C$
Observed rating  User code  Item code

However, it may lead to non-informative codes, e.g.:
1. **Unbalanced Codes**  → each bit should have split the dataset evenly
2. **Correlated Codes**  → each bit should be as independent as possible
Illustration of the effectiveness of the two constraints in DCF

Without any constraints: 3 points are (-1, -1) and 1 point is (+1, -1), which is not discriminative.

Balanced: Separated in the 1st & 3rd quadrant

Decorrelated: Well separated
However, the hard constraints of zero-mean and orthogonality may not be satisfied in Hamming space!
Our DCF Formulation

**Objective function:**

\[ \sum_{i,j \in V} (S_{ij} - b_i^T d_j)^2 + 2\alpha \| B - X \|^2 + 2\beta \| D - Y \|^2 \]

- Rating Prediction
- Constraint Trade-off

**Binary Constraint**

\[ B \in \{\pm 1\}^{r \times m}, D \in \{\pm 1\}^{r \times n} \]

**Balanced Constraint**

\[ X1 = 0, Y1 = 0, XX^T = mI, YY^T = nI \]

**Decorrelated Constraint**

**Delegate Code Quality Constraint**

**Mixed-Integer Programming NP-Hard** [Hastad 2001]
Our Solution: Alternating Optimization

Alternative Procedure

- **B-Subproblem**
  \[
  \arg\min_B \sum_{i,j \in \mathcal{V}} (S_{ij} - b_i^T d_j)^2 - 2\alpha tr(B^T X) \quad s.t., \quad B \in \{\pm 1\}^{r \times m}
  \]

- **D-Subproblem**
  \[
  \arg\min_D \sum_{i,j \in \mathcal{V}} (S_{ij} - b_i^T d_j)^2 - 2\beta tr(D^T Y) \quad s.t., \quad D \in \{\pm 1\}^{r \times n}
  \]

- **X-Subproblem**
  \[
  \arg\min_X - 2\alpha tr(B^T X) \quad s.t., \quad X1 = 0, XX^T = mI
  \]

- **Y-Subproblem**
  \[
  \arg\min_X - 2\beta tr(D^T Y) \quad s.t., \quad Y1 = 0, YY^T = nI
  \]
B-Subproblem for Binary Codes

Objective Function
\[
\arg\min_{B} \sum_{i,j \in \mathcal{V}} (S_{ij} - b_i^T d_j)^2 - 2\alpha tr(B^T X) \quad s.t., \quad B \in \{\pm 1\}^{r \times m}
\]

For each user code \( b_i \), optimize bit by bit

```
for i=1 to m do
    repeat
        for k=1 to r do
            \( \hat{b}_{ik} \leftarrow \sum_{j \in \mathcal{V}_i} (S_{ij} - d_{jk}^T b_{ik}) d_{jk} + \alpha x_{ik} \);
            \( b_{ik} \leftarrow \text{sgn}(K(\hat{b}_{ik}, b_{ik})) \);
        end
    until converge;
end
```

Parallel for loop over \( m \) users
Usually converges in 5 iterations

D-Subproblem can be solved in a similar way
B-Subproblem Complexity

\[ O(r^2 T_s \| \mathcal{V} \| / p) \]

#bits  #bit-by-bit iterations  #computing threads

#training ratings

*Linear to data size!*
**X-Subproblem for Code Delegate**

Objective Function

\[
\arg\min_x \quad -2\alpha \text{tr}(B^T X) \quad s.t., \quad X1 = 0, \quad XX^T = mI
\]

Small SVD \(r \times m\)

\[
\begin{bmatrix}
P_b \hat{P}_b \\ Q_b
\end{bmatrix} \leftarrow \text{SVD} \left( \overline{B} \right)
\]

Orthogonalization

\[
\hat{Q}_b \leftarrow \text{GramSchmidt} \left( \begin{bmatrix} Q_b & 1 \end{bmatrix} \right)
\]

\[
X \leftarrow \sqrt{m}[P_b \hat{P}_b][Q_b \hat{Q}_b]^T
\]

Y-Subproblem can be solved in a similar way
X-Subproblem Complexity

\[ O(r^2 m) \]

Linear to data size!

#bits
#users
Summary

- Recommendation is **search**
- We can accelerate search by **hashing**
- Unlike previous erroneous **two-stage** hashing, **DCF** is an **end-to-end** hashing method
- Fast **$O(n)$ discrete optimization** for DCF
Evaluations

• Dataset (filtering threshold at 10):

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Rating#</th>
<th>User#</th>
<th>Item#</th>
<th>Density</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yelp</td>
<td>696,865</td>
<td>25,677</td>
<td>25,815</td>
<td>0.11%</td>
</tr>
<tr>
<td>Amazon</td>
<td>5,057,936</td>
<td>146,469</td>
<td>189,474</td>
<td>0.02%</td>
</tr>
<tr>
<td>Netflix</td>
<td>100,480,507</td>
<td>480,189</td>
<td>17,770</td>
<td>1.18%</td>
</tr>
</tbody>
</table>

• Random split: 50% training and 50% testing.
• Metric: NDCG@K
• Search Protocol: Hamming ranking or hash table lookup
Evaluation 1: Compared to state-of-the-art

- **MF:** Matrix Factorization [Koren et al 2009]
  
  *Classific MF, Euclidean space baseline*

- **BCCF:** Binary Code learning for Collaborative Filtering
  
  [Zhou & Zha, KDD 2012]

  *MF+balance+binarization*

- **PPH:** Preference Preserving Hashing [Zhang et al. SIGIR 2014]

  *Cosine MF + norm&phase binarization*

- **CH:** Collaborative Hashing [Liu et al. CVPR 2014]

  *Full SVD MF + balance + binarization*
DCF is a new state-of-the-art

1. DCF learns compact and informative codes.
2. DCF’s performance is most close to the real-valued MF.
3. End-to-end > Two stage
Evaluation 2
DCF generalizes well to unseen users

Training: full histories of 50% users
Testing: the other 50% users that have no histories in training
Evaluation: simulate online learning scenario.

Figure 4: Recommendation performance (NDCG@10) on 50% held-out “new” users (RQ 2).
MF: original MF
MFB: MF+Binarization
DCFinit: the variant of DCF that discards the two constraints.

Figure 5: Recommendation performance (NDCG@10) of CF and DCF variants (RQ 3).
Conclusions

- **Discrete Collaborative Filtering**: an end-to-end hashing method for efficient CF
- A fast **algorithm** for DCF
- DCF is a general **framework**. It can be extended to any popular CF variants, such as SVD++ and factorization machines.
Code available: https://github.com/hanwangzhang/Discrete-Collaborative-Filtering