A two-dimensional decomposition approach for matrix completion through gossip

Mukul Bhutani and Bamdev Mishra

Problem

Approximating a matrix by decomposing it into two low rank factor matrices. Used in recommendation systems.

\[ X = U W^T \]

Challenges:
- Large matrix dims (e.g: netflix dataset: \( \sim 18000 \times 4800000 \)).
- Efficient computations needed.
- Privacy and security concerns!!

Previous Work

1. Generally treated as an optimization problem; solved using gradient search [1, 2].
2. Parallel versions of gradient search still require a central server [3, 4].
3. [5] followed \( X = U W^T \) such that each row of \( X \) and \( U \) is stored in different nodes. A public matrix \( W \) is exchanged between the nodes. Random walks are done to bring convergence in \( W \). However, here too a single agent takes care of the complete row.

Our Decomposition Pattern

1. \( X \) is decomposed into a \( p \times q \) dimensional rectangular grid of blocks.
2. Each \( X_{ij} \) can then be factored as \( U_{ij} W_{ij}^T \) as usual.
3. Each block just gossips with its neighbors and tries to reach to a consensus.
4. rows => consensus in \( U \) & each column => consensus for \( W \).
5. All these \( U \)s and \( W \)s combined together to form the universal \( U \) and \( W \).
6. This communication pattern leads to groups of blocks (\( S_{upper} \) & \( S_{lower} \)) which can be thought of as gossiping.

Problem Formulation

- Model as optimization problem. Objective function derived by analyzing \( S_{upper} \) and \( S_{lower} \).
- For \( S_{upper} \), for blocks \((i, j)\) and \((i+1, j)\) convergence in \( W_s \); for the blocks \((i, j)\) and \((i, j+1)\) \(\Rightarrow\) convergence in \( U_s \).
- Cost of a structure as comprising of two components: \( f \) and \( d \).
- \( f \Rightarrow\) measures how close it is to the original matrix.
- \( d \Rightarrow\) measures consensus between two adjacent \( U_s \)s (denoted as \( d^U \)) and \( W_s \)s (denoted as \( d^W \)).

For a structure pivoted at \((i, j)\):

\[ f_{ij} = \|X_{ij} - U_{ij} W_{ij}^T\|_F^2, \quad d_{ij}^U = \|U_{ij} - U_{ij+1}\|_F^2, \quad \text{and} \quad d_{ij}^W = \|W_{ij} - W_{ij+1}\|_F^2. \]

Consequently, the total cost \((g)\) for a structure turns out to be:

\[ g_{ij}^{upper} = f_{ij} + f_{ij+1} + f_{i+1,j} + \rho \|U_{ij} - U_{ij+1}\|_F^2 + \rho \|W_{ij} - W_{ij+1}\|_F^2, \]

where \( \rho \) is the weight factor. For \( S_{lower} \), we can derive the costs in similar fashion. For decomposition of \( X \) into \( p \times q \) end goal: minimize the sum of costs for all \( S_{upper} \)s and \( S_{lower} \)s possible, i.e.,

\[ \min_{U_{ij}, W_{ij}} \sum_{i=1, j=1}^{p,q} g_{ij}^{upper} + g_{ij}^{lower} + \lambda \|U_{ij}\|_F^2 + \lambda \|W_{ij}\|_F^2, \]

\( \lambda \) is the regularization parameter added according to [6].

Algorithm

**Algorithm 1: Basic update algorithm via SGD**

**input**: Decomposed blocks for \( X \) and rank \( r \).
**output**: \( U_s, W_s \).

\[ \begin{align*}
1. & \text{Initialize all } U_s \text{ and } W_s. \\
2. & \text{while convergence is not reached do} \\
3. & \quad S_{upper} = \text{randomly pick a valid structure.} \\
4. & \quad [U_s, W_s] = \text{updateThroughSGD}([X_s, S_{extractor}]). \\
5. & \quad \text{Check for convergence.} \\
6. & \text{end}
\end{align*} \]

Note: The number of times a particular structure may be selected is not equal for all and hence a normalization constant should be appropriately multiplied. See paper for details.

Experimentation: real datasets

RMSE on some popular datasets.

<table>
<thead>
<tr>
<th>Number of blocks ( p \times q )</th>
<th>Rank ( 2 \times 2 )</th>
<th>( 3 \times 3 )</th>
<th>( 4 \times 4 )</th>
<th>( 5 \times 5 )</th>
<th>( 10 \times 10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>MovieLens 1 million</td>
<td>5</td>
<td>0.87</td>
<td>0.99</td>
<td>1.04</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>0.86</td>
<td>0.99</td>
<td>1.03</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>0.86</td>
<td>0.99</td>
<td>1.03</td>
<td>0.99</td>
</tr>
<tr>
<td>Netflix</td>
<td>5</td>
<td>1.03</td>
<td>0.98</td>
<td>1.13</td>
<td>1.06</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1.00</td>
<td>0.98</td>
<td>1.14</td>
<td>1.02</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>1.00</td>
<td>1.11</td>
<td>1.16</td>
<td>1.02</td>
</tr>
</tbody>
</table>

Experimentation: synthetic datasets

Empirical proof of convergence of the algorithm. Cost as function of number of iterations.

<table>
<thead>
<tr>
<th>( p \times q ) (dimensions of decomposed grid)</th>
<th>( \text{Expt#1} )</th>
<th>( \text{Expt#2} )</th>
<th>( \text{Expt#3} )</th>
<th>( \text{Expt#4} )</th>
<th>( \text{Expt#5} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td></td>
</tr>
<tr>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td>( 500 \times 500 )</td>
<td></td>
</tr>
<tr>
<td>( 5 \times 5 )</td>
<td>( 6 \times 6 )</td>
<td>( 5 \times 5 )</td>
<td>( 5 \times 5 )</td>
<td>( 5 \times 5 )</td>
<td></td>
</tr>
<tr>
<td>NumIterations</td>
<td>( \text{Exp1} )</td>
<td>( \text{Exp2} )</td>
<td>( \text{Exp3} )</td>
<td>( \text{Exp4} )</td>
<td>( \text{Exp5} )</td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>0</td>
<td>1.45e+05</td>
<td>1.45e+05</td>
<td>1.45e+05</td>
<td>1.44e+05</td>
<td>6.42e+05</td>
</tr>
<tr>
<td>80000</td>
<td>6.92e+03</td>
<td>1.32e+01</td>
<td>1.40e+00</td>
<td>1.47e+02</td>
<td>1.26e+05</td>
</tr>
<tr>
<td>160000</td>
<td>9.46e+06</td>
<td>7.65e+05</td>
<td>1.44e+04</td>
<td>9.94e+05</td>
<td>2.53e+06</td>
</tr>
<tr>
<td>240000</td>
<td>1.07e+05</td>
<td>1.25e+05</td>
<td>1.04e+02</td>
<td>2.58e+03</td>
<td>8.62e+05</td>
</tr>
<tr>
<td>260000</td>
<td>1.21e+05</td>
<td>4.41e+05</td>
<td>7.99e-02</td>
<td>7.36e-02</td>
<td></td>
</tr>
<tr>
<td>280000</td>
<td>1.93e+05</td>
<td>9.28e+05</td>
<td>2.09e-02</td>
<td>9.24e-03</td>
<td></td>
</tr>
<tr>
<td>300000</td>
<td>2.09e+05</td>
<td>6.44e-03</td>
<td>6.44e-03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>400000</td>
<td>convergence</td>
<td>convergence</td>
<td>convergence</td>
<td>convergence</td>
<td>convergence</td>
</tr>
</tbody>
</table>

References