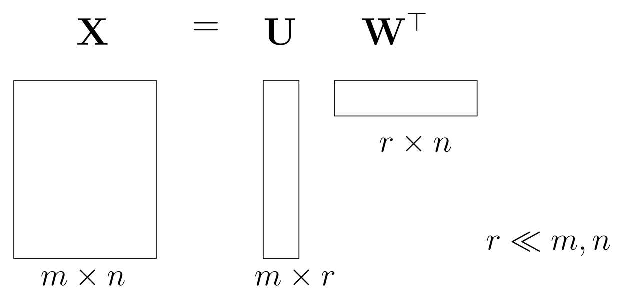
A two-dimensional decomposition approach for matrix completion through gossip

Mukul Bhutani and Bamdev Mishra

Problem

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



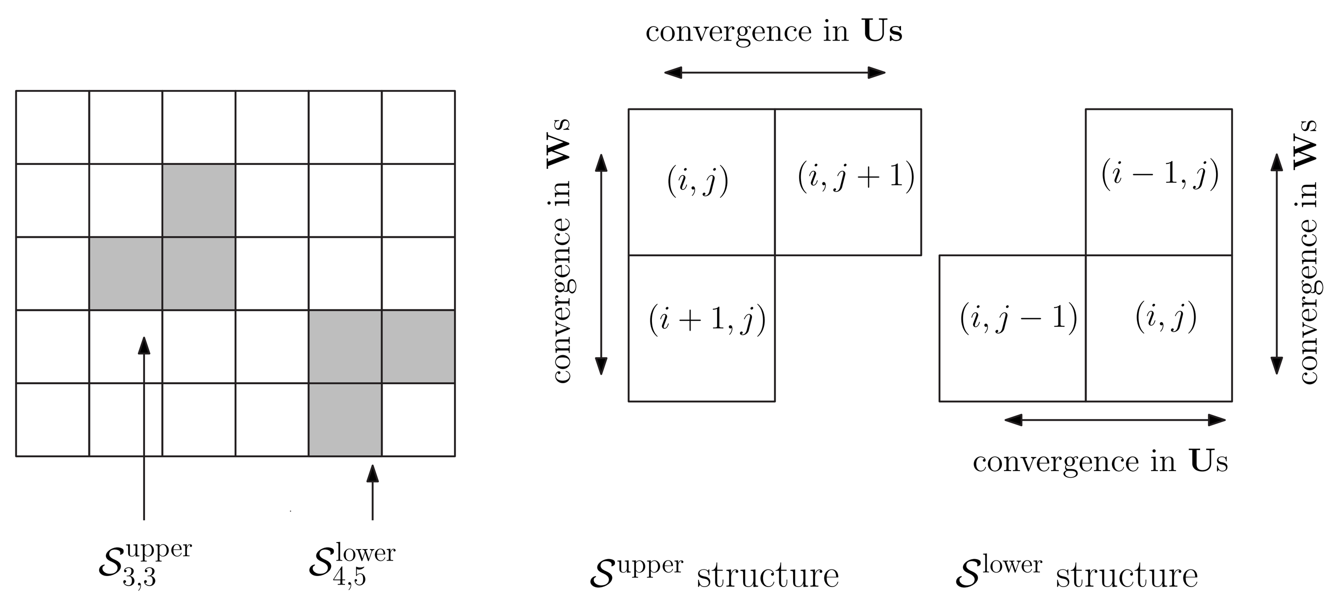
Challenges:

- Large matrix dims (e.g: netflix dataset: $\sim 18000 \times 480000$).
- 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 $\mathbf{X} = \mathbf{U}\mathbf{W}^\top$ such that each row of $\mathbf{X}$ and $\mathbf{U}$ is stored in different nodes. A public matrix $\mathbf{W}$ is exchanged between the nodes. Random walks are done to bring convergence in $\mathbf{W}$. However, here too a single agent takes care of the complete row.

Our Decomposition Pattern



Example Decomposition

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

Problem Formulation

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

For a structure pivoted at (i, j):

$$f_{ij} = \left\|\mathbf{X}_{ij} - \mathbf{U}_{ij}{\mathbf{W}^\top}_{ij}\right\|_F^2, d^U_{ij} = \|\mathbf{U}_{ij} - \mathbf{U}_{ij+1}\|_F^2, \text{ and } d^W_{ij} = \|\mathbf{W}_{ij} - \mathbf{W}_{i+1j}\|_F^2,$$

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

$$g^{\text{upper}}_{ij} = f_{ij} + f_{i+1j} + f_{ij+1} + \rho \|\mathbf{U}_{ij} - \mathbf{U}_{ij+1}\|_F^2 + \rho \|\mathbf{W}_{ij} - \mathbf{W}_{i+1j}\|_F^2.$$

where ρ is the *weight* factor. For $\mathcal{S}^{\text{lower}}$, we can derive the costs in similar fashion. For decomposition of $\mathbf{X}$ into $p \times q$ end goal: minimize the sum of costs for all $\mathcal{S}^{\text{upper}}$and $\mathcal{S}^{\text{lower}}$possible, i.e.,

$$\min_{\mathbf{U}_{ij}, \mathbf{W}_{ij}} \sum_{i=1, j=1}^{p,q} g^{\text{upper}}_{ij} + g^{\text{lower}}_{ij} + \lambda \|\mathbf{U}_{ij}\|_F^2 + \lambda \|\mathbf{W}_{ij}\|_F^2,$$

λ is the regularization parameter added according to [6].

Algorithm

Algorithm 1: Basic update algorithm via SGD

input : Decomposed blocks for $\mathbf{X}$ and rank r.
output: $\mathbf{U}$s, $\mathbf{W}$s.

```
Initialize all Us and Ws.
while convergence is not reached do
    S^struct = randomly pick a valid structure.
    [Us, Ws] = updateThroughSGD(Xs, S^struct).
    Check for convergence.
end
```

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.

	Number of blocks $p \times q$				
Rank	2×2	3×3	4×4	5×5	10×10
			MovieLens 1 million		
5	0.87	0.99	1.04	0.99	1.13
10	0.86	0.99	1.03	1.00	1.22
15	0.86	0.99	1.03	0.99	1.34
			MovieLens 20 million		
5	0.95	0.92	0.93	0.99	1.01
10	0.96	0.93	0.93	1.02	1.11
15	0.96	0.94	0.93	1.05	1.24
			Netflix		
5	1.03	0.98	1.13	1.06	1.02
10	1.00	0.98	1.14	1.02	1.02
15	1.00	1.11	1.16	1.02	1.03

Experiments: synthetic datasets

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

$m \times n$ (input matrix dimensions) $p \times q$ (dimensions of decomposed grid)	Exp#1 500×500 4×4	Exp#2 500×500 4×5	Exp#3 500×500 5×5	Exp#4 500×500 6×6	Exp#5 5000×5000 5×5
NumIterations					
0	1.45e+05	1.45e+05	1.45e+05	1.44e+05	6.42e+05
80000	6.92e-03	1.32e-01	1.45e+00	4.74e+02	1.26e+05
160000	9.62e-06	7.65e-05	1.44e-04	9.94e-01	2.83e+02
240000	convergence	1.07e-05	1.25e-05	1.04e-02	2.85e-01
260000		convergence	1.21e-05	4.41e-03	7.39e-02
280000			convergence	1.96e-03	2.09e-02
300000				9.28e-04	6.44e-03
400000				convergence	convergence

References

[1] Genevieve Gorrell. Generalized hebbian algorithm for incremental singular value decomposition in natural language processing. In *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2006.
[2] Naiyang Guan, Dacheng Tao, Zhigang Luo, and Bo Yuan. Nenmf: An optimal gradient method for nonnegative matrix factorization. *IEEE Transactions on Signal Processing 60*, 6:2882–2898, 2012.
[3] Cheng-Tao Chu, Sang Kyun Kim, Yi-An Lin, YuanYuan Yu, Gary Bradski, Andrew Ng, and Kunle Olukotun. Map-reduce for machine learning on multicore. In *Neural Information Processing Systems (NIPS)*, pages 281–288, 2006.
[4] Quoc Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg Corrado, Jeff Dean, and Andrew Ng. Building high-level features using large scale unsupervised learning. In *International Conference in Machine Learning*, pages 81–88, 2012.
[5] István Hegedűs, Márk Jelasity, Levente Kocsis, and András A. Benczúr. Fully distributefully distributed robust singular value decomposition. In *14-th IEEE International Conference on Peer-to-Peer Computing (P2P)*, 2014.
[6] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning. Technical report, arXiv preprint arXiv:1606.04838, 2016.