Geometric Matrix Completion with Recurrent Multi-Graph Neural Networks

Credit: Federico Monti[1], Michael M. Bronstein[1], Xavier Bresson[2]

[1]Universita della Svizzera italiana

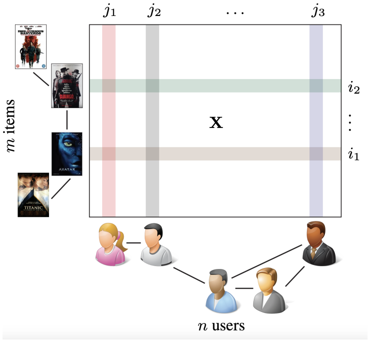
[2]Nanyang Technological University

Presenter: Fuwen Tan

https://qdata.github.io/deep2Read

Problem

- $X \in \mathbb{R}^{m \times n}$
- X_{ij}: the probability that USER j likes ITEM i
- Goal: reconstruct a huge X from a sparse set of known $\{X_{ij}\}$
- Example: Netflix challege [8], 480k movies $\times$ 18k users (8.5B entries), with only 0.011% known entries.



Previous effort 1: low rank assumption

- Key assumption: low rank (each entry is defined by a (USER, ITEM) pair, many entries share the same USER or ITEM)
- Challenge: NP-hard combinatorial problem
- Ω: known entries set

$$\min_{\mathbf{X}} \ \mathrm{rank}(\mathbf{X}) \quad \text{s.t.} \quad x_{ij} = y_{ij}, \ \forall ij \in \Omega, \tag{1}$$

Previous effort 2: convex solution

- Convex version
- $\|\cdot\|_\star$: nuclear norm, L1-norm of the eigenvalue vector
- "Under some technical conditions" [1], (2) has the same solution as (1).

$$\min_{\mathbf{X}} \ \|\mathbf{X}\|_\star + \frac{\mu}{2}\|\mathbf{\Omega} \circ (\mathbf{X} - \mathbf{Y})\|_{\mathrm{F}}^2, \tag{2}$$

Previous effort 3: geometric method

- Model the relationship among the USERs
  - $\mathcal{G}_c = (\{1, \ldots, n\}, \mathcal{E}_c, \mathbf{W}_c)$, $\mathcal{E}_c$: edge (relation) set.
  - $\mathbf{W}_c = (w^c_{ij})$: *adjacency matrix*
  - $\mathbf{\Delta}_c = \mathbf{I} - \mathbf{D}^{-1/2}\mathbf{W}_c\mathbf{D}^{-1/2}$: graph Laplacian
  - $\|\mathbf{X}\|^2_{\mathcal{G}_c} = \mathrm{trace}(\mathbf{X}\mathbf{\Delta}_c\mathbf{X}^\top)$: *Dirichlet norm*
  - Both $\mathbf{X}$ and $\mathbf{\Delta}_c\mathbf{X}^\top$ should be small.
- Model the relationship among the ITEMs
  - Similiar $\mathcal{G}_r, \mathbf{W}_r, \mathcal{E}_r$, $\|\mathbf{X}\|^2_{\mathcal{G}_r} = \mathrm{trace}(\mathbf{X}^\top\mathbf{\Delta}_r\mathbf{X})$

$$\min_{\mathbf{X}} \ \|\mathbf{X}\|^2_{\mathcal{G}_r} + \|\mathbf{X}\|^2_{\mathcal{G}_c} + \frac{\mu}{2}\|\mathbf{\Omega} \circ (\mathbf{X} - \mathbf{Y})\|^2_{\mathrm{F}}, \tag{3}$$

Previous effort 4: factorized models

- Explicitly model the low rank assumption by a factorized representation [12]
- $\mathbf{X} = \mathbf{W}\mathbf{H}^\top$
- $\mathbf{W}, \mathbf{H}$ are $m \times r$ and $n \times r$ matrices, $r \ll \min(m, n)$
- So, $\mathrm{rank}(\mathbf{X}) \ll \min(m, n)$
- $\| \cdot \|_{\mathrm{F}}^2$: Frobenius norm (L2-norm?)

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2}\|\mathbf{W}\|_{\mathrm{F}}^2 + \frac{1}{2}\|\mathbf{H}\|_{\mathrm{F}}^2 + \frac{\mu}{2}\|\mathbf{\Omega} \circ (\mathbf{W}\mathbf{H}^\top - \mathbf{Y})\|_{\mathrm{F}}^2. \tag{4}$$

$$\min_{\mathbf{W},\mathbf{H}} \frac{1}{2}\|\mathbf{W}\|_{\mathcal{G}_r}^2 + \frac{1}{2}\|\mathbf{H}\|_{\mathcal{G}_c}^2 + \frac{\mu}{2}\|\mathbf{\Omega} \circ (\mathbf{W}\mathbf{H}^\top - \mathbf{Y})\|_{\mathrm{F}}^2. \tag{5}$$

Recall: spectral graph convolution

- (6): eigen decomposition of the graph Laplacian
- (7): spectral convolution between signal x and filter y
- (8): spectral convolution layer
  - Input $\mathbf{x} = \{\mathbf{x}_{l'}\} \in \mathbb{R}^{h \times w \times q'}$; Output $\tilde{\mathbf{x}} = \{\mathbf{x}_l\} \in \mathbb{R}^{h \times w \times q}$
- Drawbacks:
  - $O(n)$ parameters, no weight sharing
  - $O(n^2)$ computations (multiplication with $\mathbf{\Phi}$)

$$\mathbf{\Delta} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^\top \tag{6}$$

$$\mathbf{x} \star \mathbf{y} = \mathbf{\Phi}(\mathbf{\Phi}^\top \mathbf{x}) \circ (\mathbf{\Phi}^\top \mathbf{y}) = \mathbf{\Phi}\, \mathrm{diag}(\hat{y}_1, \ldots, \hat{y}_n)\, \hat{\mathbf{x}} \tag{7}$$

$$\tilde{\mathbf{x}}_l = \xi \left(\sum_{l'=1}^{q'} \mathbf{\Phi} \hat{\mathbf{Y}}_{ll'} \mathbf{\Phi}^\top \mathbf{x}_{l'} \right), \quad l = 1, \ldots, q \tag{8}$$

Recall: CNN on Graphs with Fast Localized Spectral Filtering [3]

- Chebyshev polynomial:
  $T_0(\lambda) = 1$, $T_1(\lambda) = \lambda$, $T_j(\lambda) = 2\lambda T_{j-1}(\lambda) - T_{j-2}(\lambda)$
- Rescaled graph Laplacian: $\tilde{\mathbf{\Delta}} = 2\lambda_n^{-1}\mathbf{\Delta} - \mathbf{I}$
- Rescaled eigen values: $\tilde{\mathbf{\Lambda}} = 2\lambda_n^{-1}\mathbf{\Lambda} - \mathbf{I}$
- Benefits:
  - $p = O(1)$ parameters
  - $O(n)$ computations (no multiplication with $\mathbf{\Phi}$)

$$\tilde{\mathbf{x}} = \tau_{\boldsymbol{\theta}}(\tilde{\mathbf{\Delta}})\mathbf{x} \tag{9}$$

$$\tau_{\boldsymbol{\theta}}(\tilde{\mathbf{\Delta}}) = \sum_{j=0}^{p-1} \theta_j \mathbf{\Phi} T_j(\tilde{\mathbf{\Lambda}}) \mathbf{\Phi}^\top = \sum_{j=0}^{p-1} \theta_j T_j(\tilde{\mathbf{\Delta}}) \tag{10}$$

Separable CNNs (sMGCNN)

If the factorized representation is used:

$$\tilde{\mathbf{w}}_l = \xi(\sum_{l'=1}^{q'} \sum_{j=0}^{p} \theta^r_{ll',j} T_j(\tilde{\boldsymbol{\Delta}}_r)\mathbf{w}_{l'}) \tag{11}$$

$$\tilde{\mathbf{h}}_l = \xi(\sum_{l'=1}^{q'} \sum_{j'=0}^{p} \theta^c_{ll',j'} T_{j'}(\tilde{\boldsymbol{\Delta}}_c)\mathbf{h}_{l'}) \tag{12}$$

Multi-Graph CNNs (MGCNN)

Multi-Graph spectral convolution

$$\hat{\mathbf{X}} = \mathbf{\Phi}_r^\top \mathbf{X} \mathbf{\Phi}_c \tag{13}$$

$$\mathbf{X} \star \mathbf{Y} = \mathbf{\Phi}_r (\hat{\mathbf{X}} \circ \hat{\mathbf{Y}}) \mathbf{\Phi}_c^\top . \tag{14}$$

Multi-Graph CNNs (MGCNN)

Localized version

$$\tilde{\mathbf{X}} = \sum_{j,j'=0}^{p} \theta_{jj'} T_j(\tilde{\boldsymbol{\Delta}}_r) \mathbf{X} T_{j'}(\tilde{\boldsymbol{\Delta}}_c) \qquad (15)$$

- $(p+1)^2 = O(1)$ parameters
- $O(mn)$ computations, linear complexity

Matrix diffusion with RNNs

Algorithm 1 (RMGCNN)

input $m \times n$ matrix $\mathbf{X}^{(0)}$ containing initial values

1: **for** $t = 0 : T$ **do**
2: Apply the Multi-Graph CNN (13) on $\mathbf{X}^{(t)}$ producing an $m \times n \times q$ output $\tilde{\mathbf{X}}^{(t)}$.
3: **for** all elements (i, j) **do**
4: Apply RNN to q-dim $\tilde{\mathbf{x}}_{ij}^{(t)} = (\tilde{x}_{ij1}^{(t)}, \dots, \tilde{x}_{ijq}^{(t)})$ producing incremental update $dx_{ij}^{(t)}$
5: **end for**
6: Update $\mathbf{X}^{(t+1)} = \mathbf{X}^{(t)} + \mathbf{dX}^{(t)}$
7: **end for**

Algorithm 2 (sRMGCNN)

input $m \times r$ factor $\mathbf{H}^{(0)}$ and $n \times r$ factor $\mathbf{W}^{(0)}$ representing the matrix $\mathbf{X}^{(0)}$

1: **for** $t = 0 : T$ **do**
2: Apply the Graph CNN on $\mathbf{H}^{(t)}$ producing an $n \times q$ output $\tilde{\mathbf{H}}^{(t)}$.
3: **for** $j = 1 : n$ **do**
4: Apply RNN to q-dim $\tilde{\mathbf{h}}_{j}^{(t)} = (\tilde{h}_{j1}^{(t)}, \dots, \tilde{h}_{jq}^{(t)})$ producing incremental update $dh_{j}^{(t)}$
5: **end for**
6: Update $\mathbf{H}^{(t+1)} = \mathbf{H}^{(t)} + \mathbf{dH}^{(t)}$
7: Repeat steps 2-6 for $\mathbf{W}^{(t+1)}$
8: **end for**

(Separable) Recurrent Multi-Graph CNN

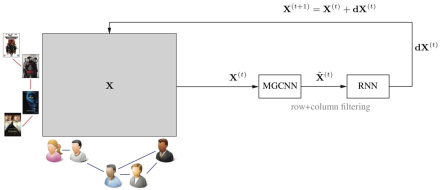


Figure: Recurrent MGCNN (RMGCNN)

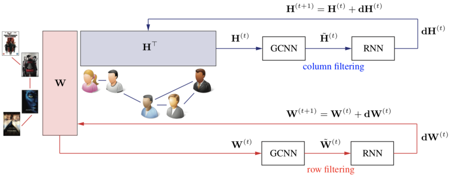


Figure: Separable Recurrent MGCNN (sRMGCNN)

Loss

Full matrix representation:

$$\ell(\boldsymbol{\Theta}, \boldsymbol{\sigma}) = \|\mathbf{X}^{(T)}_{\boldsymbol{\Theta},\boldsymbol{\sigma}}\|^2_{\mathcal{G}_r} + \|\mathbf{X}^{(T)}_{\boldsymbol{\Theta},\boldsymbol{\sigma}}\|^2_{\mathcal{G}_c} + \frac{\mu}{2}\|\boldsymbol{\Omega} \circ (\mathbf{X}^{(T)}_{\boldsymbol{\Theta},\boldsymbol{\sigma}} - \mathbf{Y})\|^2_{\mathrm{F}}. \tag{16}$$

Factorized representation:

$$\ell(\boldsymbol{\theta}_r, \boldsymbol{\theta}_c, \boldsymbol{\sigma}) = \|\mathbf{W}^{(T)}_{\boldsymbol{\theta}_r,\boldsymbol{\sigma}}\|^2_{\mathcal{G}_r} + \|\mathbf{H}^{(T)}_{\boldsymbol{\theta}_c,\boldsymbol{\sigma}}\|^2_{\mathcal{G}_c} + \frac{\mu}{2}\|\boldsymbol{\Omega} \circ (\mathbf{W}^{(T)}_{\boldsymbol{\theta}_r,\boldsymbol{\sigma}}(\mathbf{H}^{(T)}_{\boldsymbol{\theta}_c,\boldsymbol{\sigma}})^\top - \mathbf{Y})\|^2_{\mathrm{F}}.$$

Experiments

- $p = 4, T = 10, q = 32$
- Datasets:
  - Synthetic data [7]
  - MovieLens [10]
  - Flixster [6]
  - Douban [9]
  - YahooMusic [4]
- Baselines:
  - Classical Matrix Completion (MC) [2]
  - Inductive Matrix Completion (IMC) [5]
  - Geometric Matrix Completion (GMC) [7]
  - Graph Regularized Alternating Least Squares (GRALS) [11]

Synthetic data

Table: Comparison of different matrix completion methods using *users+items graphs* in terms of number of parameters (optimization variables) and computational complexity order (operations per iteration). Rightmost column shows the RMS error on Synthetic dataset.

METHOD	PARAMETERS	COMPLEXITY	RMSE
GMC	$O(mn)$	$O(mn)$	0.3693
GRALS	$O(m+n)$	$O(m+n)$	0.0114
RGCNN	$\mathbf{O}(\mathbf{1})$	$\mathbf{O}(\mathbf{mn})$	**0.0053**
sRGCNN	$\mathbf{O}(\mathbf{1})$	$\mathbf{O}(\mathbf{m}+\mathbf{n})$	**0.0106**

Synthetic data

Table: Comparison of different matrix completion methods using *users graph only* in terms of number of parameters (optimization variables) and computational complexity order (operations per iteration). Rightmost column shows the RMS error on Synthetic dataset.

Method	Parameters	Complexity	RMSE
GRALS	$O(m+n)$	$O(m+n)$	0.0452
sRGCNN	$\mathbf{O}(\mathbf{m})$	$\mathbf{O}(\mathbf{m}+\mathbf{n})$	**0.0362**

Synthetic data

Table 3: Reconstruction errors for the synthetic dataset between multiple convolutional layers architectures and the proposed architecture. Chebyshev polynomials of order 4 have been used for both users and movies graphs (q'MGCq denotes a multi-graph convolutional layer with q' input features and q output features).

Method	Params	Architecture	RMSE
$\text{MGCNN}_{\text{3layers}}$	$9K$	1MGC32, 32MGC10, 10MGC1	0.0116
$\text{MGCNN}_{\text{4layers}}$	$53K$	1MGC32, 32MGC32 $\times$ 2, 32MGC1	0.0073
$\text{MGCNN}_{\text{5layers}}$	$78K$	1MGC32, 32MGC32 $\times$ 3, 32MGC1	0.0074
$\text{MGCNN}_{\text{6layers}}$	$104K$	1MGC32, 32MGC32 $\times$ 4, 32MGC1	0.0064
RMGCNN	**9K**	1MGC32 + LSTM	**0.0053**

Real data

Table: Performance (RMS error) of different matrix completion methods on the MovieLens dataset.

Method	RMSE
Global Mean	1.154
User Mean	1.063
Movie Mean	1.033
MC [2]	0.973
IMC [5]	1.653
GMC [7]	0.996
GRALS [11]	0.945
sRGCNN	**0.929**

Real data

Table: Matrix completion results on several datasets (RMS error). For Douban and YahooMusic, a single graph (of users and items, respectively) was used. For Flixter, two settings are shown: users+items graphs / only users graph.

METHOD	FLIXSTER	DOUBAN	YAHOOMUSIC
GRALS	1.3126 / 1.2447	0.8326	38.0423
sRGCNN	**1.1788** / **0.9258**	**0.8012**	**22.4149**

Conclusion

- Application of [3] on Matrix Completion problems
- Open source
- Extensive experiments

Emmanuel J. Candès and Benjamin Recht.
Exact matrix completion via convex optimization.
Foundations of Computational Mathematics, 9(6):717, Apr 2009.

Emmanuel J. Candès and Benjamin Recht.
Exact matrix completion via convex optimization.
Commun. ACM, 55:111–119, 2012.

Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst.
Convolutional neural networks on graphs with fast localized spectral filtering.
In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 3844–3852. Curran Associates, Inc., 2016.

Gideon Dror, Noam Koenigstein, Yehuda Koren, and Markus Weimer.
The yahoo! music dataset and kdd-cup '11.
In *KDD Cup*, 2011.

Prateek Jain and Inderjit S. Dhillon.

Provable inductive matrix completion.
CoRR, abs/1306.0626, 2013.

Mohsen Jamali and Martin Ester.
A matrix factorization technique with trust propagation for recommendation in social networks.
In *RecSys*, 2010.

Vassilis Kalofolias, Xavier Bresson, Michael M. Bronstein, and Pierre Vandergheynst.
Matrix completion on graphs.
CoRR, abs/1408.1717, 2014.

Y. Koren, R. Bell, and C. Volinsky.
Matrix factorization techniques for recommender systems.
Computer, 42(8):30–37, Aug 2009.

Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King.
Recommender systems with social regularization.
In *WSDM*, 2011.

Bradley N. Miller, Istvan Albert, Shyong K. Lam, Joseph A. Konstan, and John Riedl.
Movielens unplugged: experiences with an occasionally connected recommender system.
In *IUI*, 2003.

Nikhil Rao, Hsiang-Fu Yu, Pradeep Ravikumar, and Inderjit S. Dhillon.
Collaborative filtering with graph information: Consistency and scalable methods.
In *NIPS*, 2015.

Nathan Srebro, Jason Rennie, and Tommi S. Jaakkola.
Maximum-margin matrix factorization.
In *Advances in Neural Information Processing Systems (NIPS)*, pages 1329–1336. 2005.