Geometric Matrix Completion with Recurrent Multi-Graph Neural Networks

Federico Monti, Michael M. Bronstein, and Xavier Bresson

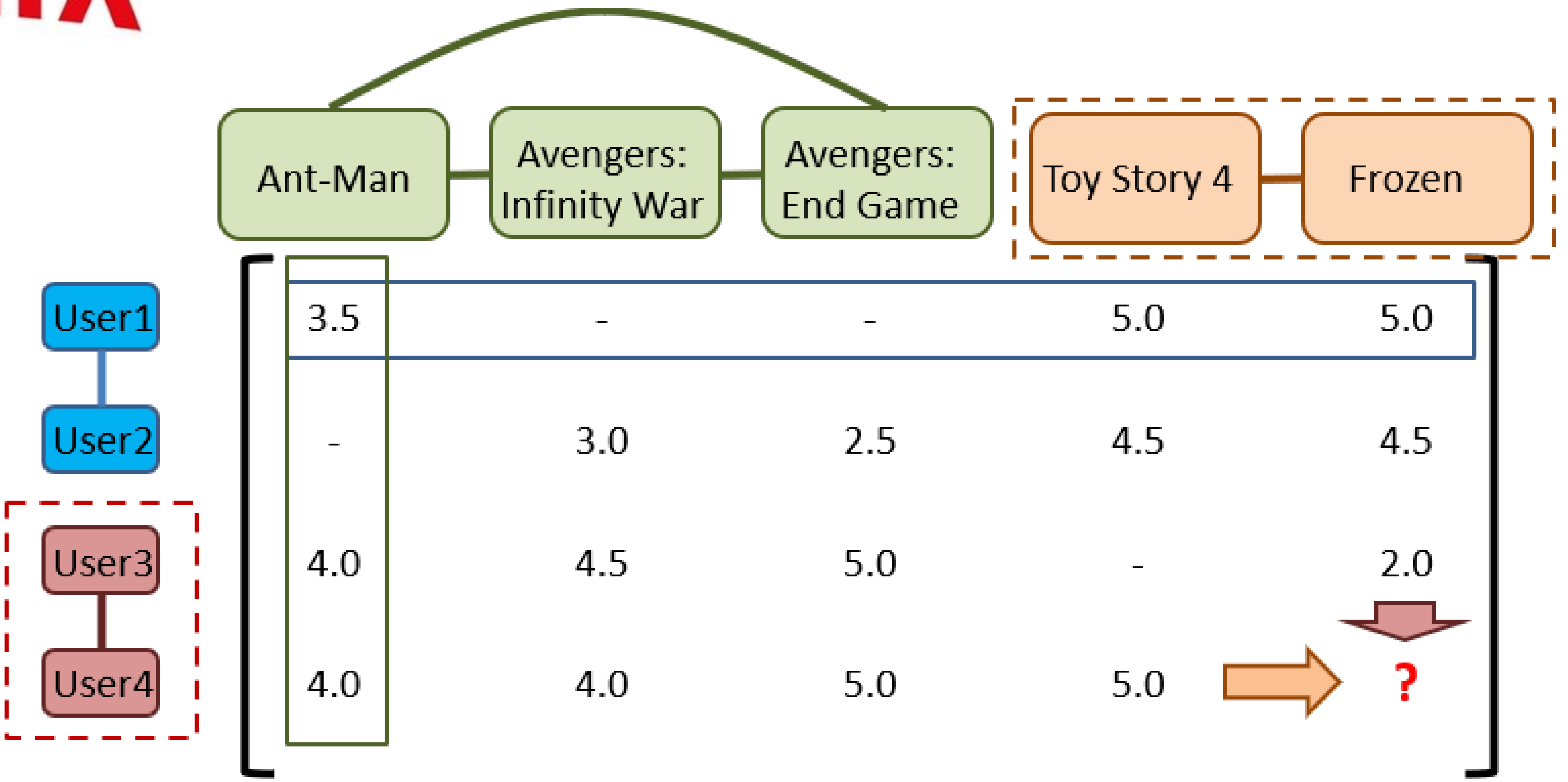
NIPS 2017

Presenter : Mineui Hong

Recommendation & Matrix Completion

NETFLIX

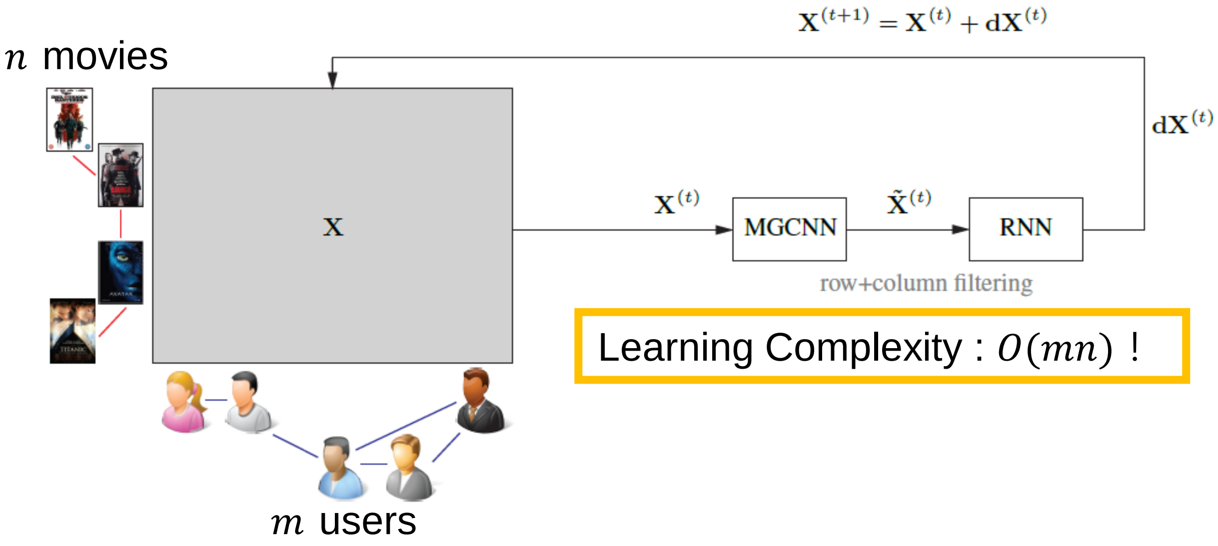
Row graph + Column graph -> Multi-Graph Convolution!



Multi-Graph Convolutional Neural Network (MGCNN)

	1-Dimensional	2-Dimensional
Graph Laplacian	$\Delta = \Phi\Lambda\Phi^T$ $\Lambda = \mathrm{diag}(\lambda_1, \cdots, \lambda_n)$ Φ : eigenvector matrix	$\Delta_c = \Phi_c\Lambda_c\Phi_c^T,$ $\Delta_r = \Phi_r\Lambda_r\Phi_r^T$
Fourier Transform	$\hat{X} = \Phi^T X$	$\hat{X} = \Phi_r^T X \Phi_c$
Convolution Operation	$X * Y = \Phi(\Phi^T X) \circ (\Phi^T Y)$ $= \Phi(\hat{X} \circ \hat{Y})$	$X * Y = \Phi_r(\hat{X} \circ \hat{Y})\Phi_c^T$
Chebychev Polynomial Filter	$\tau_\theta(\lambda) = \sum_{j=0}^{p} \theta_j T_j(\tilde{\lambda}),$ $\tilde{\lambda} \in [-1,1]$	$\tau_\Theta(\lambda_c, \lambda_r) = \sum_{j,j'=0}^{p} \theta_{j,j'} T_j(\tilde{\lambda}_c)\, T_{j'}(\tilde{\lambda}_r)$
Graph CNN	$\tilde{X}_l = \xi\left(\sum_{l'=1}^{q'} \sum_{j=0}^{p} \theta_{j,ll'} T_j(\tilde{\Delta}) X_l\right)$ l', l : input / output channel ξ : nonlinearity function	$\tilde{X}_l = \xi\left(\sum_{l'=1}^{q'} \sum_{j,j'=0}^{p} \theta_{jj',ll'} T_j(\tilde{\Delta}_r) X_{l'} T_{j'}(\tilde{\Delta}_c)\right)$

Recurrent MGCNN (RMGCNN)



Learning Complexity : $O(mn)$!

Matrix Diffusion :
Make small changes for each step

Loss :

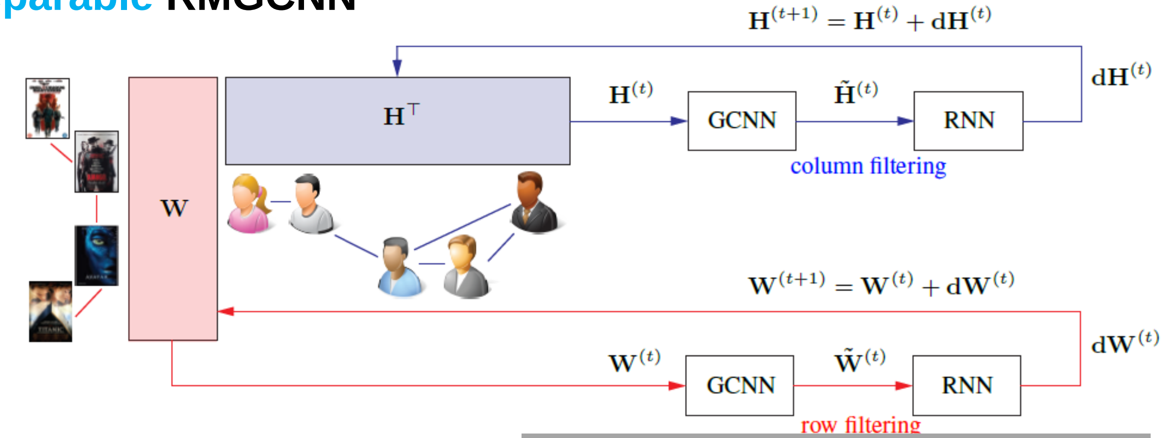
Smoothness on column-graph

$$\left\|X_{\Theta,\sigma}^{(T)}\right\|_{g_r}^2 + \left\|X_{\Theta,\sigma}^{(T)}\right\|_{g_c}^2 + \frac{\mu}{2}\left\|\Omega \circ \left(X_{\Theta,\sigma}^{(T)} - Y\right)\right\|_F^2$$

Smoothness on row-graph

Error on labeled data

Separable RMGCNN



Learning Complexity : $O(n+m)$

Matrix Factorization :

$$X = WH^T : n \times m$$

$$W : n \times r, \quad H : m \times r \quad (r \ll n, m)$$

Loss :

$$\left\|W_{\Theta_r,\sigma}^{(T)}\right\|_{g_r}^2 + \left\|H_{\Theta_c,\sigma}^{(T)}\right\|_{g_c}^2 + \frac{\mu}{2}\left\|\Omega \circ \left(W_{\Theta_r,\sigma}^{(T)}\left(H_{\Theta_c,\sigma}^{(T)}\right)^{\mathrm{T}} - Y\right)\right\|_F^2$$

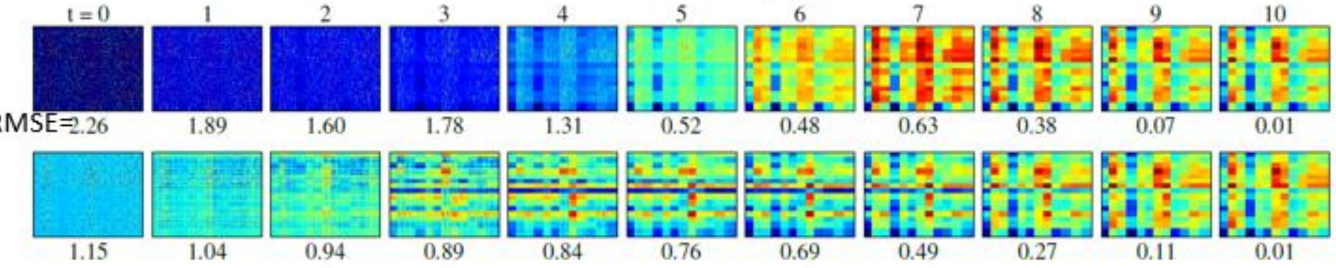
Experimental Results

- Evolution of Matrix (RMS error heatmap)

MGCNN

Mean RMSE=

sMGCNN



- Synthetic Data

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

- MovieLens Dataset

METHOD	RMSE
GLOBAL MEAN	1.154
USER MEAN	1.063
MOVIE MEAN	1.033
MC [9]	0.973
IMC [17, 42]	1.653
GMC [19]	0.996
GRALS [33]	0.945
sRMGCNN	**0.929**

Conclusion

- Generalize a graph convolutional network into
  a **multi-graph convolutional network**

- Handle the matrix completion problem with
  multi-graph CNN + **matrix diffusion** using RNN structure

- **Simple & efficient** algorithm,
  seems to be practical to apply on various large-scale applications.

Thank You