GraphCAR: Content-aware Multimedia Recommendation with Graph Autoencoder

Qidi Xu
Center for Future Media, University of Electronic Science and Technology of China
qidixu1996@gmail.com

Fumin Shen
Center for Future Media, University of Electronic Science and Technology of China
fumin.shen@gmail.com

Li Liu
School of Computing Sciences, University of East Anglia
li.liu@uea.ac.uk

Heng Tao Shen
Center for Future Media, University of Electronic Science and Technology of China
shenhengtao@hotmail.com

ABSTRACT

Precisely recommending relevant multimedia items from massive candidates to a large number of users is an indispensable yet difficult task on many platforms. A promising way is to project users and items into a latent space and recommend items via the inner product of latent factor vectors. However, ~~previous studies paid little attention to the multimedia content~~ itself and couldn't make the best use of preference data like implicit feedback. To fill this gap, we propose a Content-aware Multimedia Recommendation Model with Graph Autoencoder (GraphCAR), combining informative multimedia content with user-item interaction. Specifically, user-item interaction, user attributes and multimedia contents (*e.g.*, images, videos, audios, *etc.*) are taken as input of the autoencoder to generate the item preference scores for each user. Through extensive experiments on two real-world multimedia Web services: Amazon and Vine, we show that GraphCAR significantly outperforms state-of-the-art techniques of both collaborative filtering and content-based methods.

CCS CONCEPTS

• **Information systems → Multimedia information systems**; • **Computing methodologies → Visual content-based indexing and retrieval**;

KEYWORDS

Multimedia Recommendation, Graph Autoencoder, Content-based Filtering, Implicit Feedback

ACM Reference Format:
Qidi Xu, Fumin Shen, Li Liu, and Heng Tao Shen. 2018. GraphCAR: Content-aware Multimedia Recommendation with Graph Autoencoder. In *SIGIR '18: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, July 8-12, 2018, Ann Arbor, MI, USA*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3209978.3210117

SIGIR'18, July 8–12, 2018, Ann Arbor, MI, USA

ACM ISBN 978-1-4503-5657-2/18/07…$15.00
https://doi.org/10.1145/3209978.3210117

1 INTRODUCTION

In recent years, recommender systems have played a significant part in the multimedia field. However, within most of these web services, the number of users and the number of images/videos are dramatically growing, making the multimedia recommendation more challenging than ever before. The dominating Web multimedia content requires modern recommender systems, in particular, those based on Collaborative Filtering (CF), to sift through massive multimedia content for users in a highly dynamic environment.

Collaborative filtering methods group people with similar interests and make recommendations on this basis. In the context of multimedia recommendation, *item* indicates different kinds of multimedia content. Most CF methods rely on items' star ratings, which provide *explicit feedback* [5, 9].

However, when used in the multimedia field, traditional CF methods have two shortcomings. First, CF methods failed to focus on the multimedia content itself, which is the most important factor when users choose images or videos. As content information of items is available, content-aware methods have been introduced. Such incorporation of content information usually leads to better recommendation performance [1, 10, 12]. Second, explicit ratings are not always available in many applications. More often, interaction data such as "like" of photos, or "view" of movies are more convenient to collect. Such data are based on *implicit feedback*.

这个问题本文怎么解决的？

To combine the multimedia content with the CF method and make the best use of implicit feedback, we propose a novel content-aware multimedia recommendation framework with graph autoencoder (GraphCAR). We use two graph convolutional networks as the encoder to model latent factor of users and items, respectively. After that, we generate preference scores by using the inner product of two latent factor vectors.

We evaluate GraphCAR extensively on two real-world datasets that represent a spectrum of different media: the Amazon Movies and TVs dataset and the Vine dataset, with the former providing images features and the latter providing videos features. Through these experiments, we observe that GraphCAR is superior to competing methods of the best configuration, ranging from CF-based methods to content-based methods.

In summary, the main contributions of our work are:

- We propose a novel content-aware multimedia recommendation model with graph autoencoder (GraphCAR) to employ graph autoencoder in CF with implicit feedback.
- To combine user-item interaction with user attributes and the multimedia content, we introduce two graph convolutional networks, both of which are neural networks that can be seamlessly incorporated into any neighborhood models with efficient end-to-end SGD training.
- Experiments on two real datasets show that GraphCAR significantly outperforms state-of-the-art techniques of both CF and content-based methods.

2 PRELIMINARIES

Before we dive into the details of our models, we first discuss the preliminaries as follows.

2.1 Problem Formulation

Given users $u = [1, 2, ..., N]$, items $v = [1, 2, ..., M]$, and the interaction matrix $R \subseteq \mathbb{R}^{N \times M}$, R_{ij} indicates the interaction between the i-th user u_i and the j-th item v_j. We use $\mathcal{R} = \{(i, j) | R_{ij} = 1\}$ to denote the user-item pairs where there exist implicit interactions. The goal of our work is to predict the potential items with which users may likely to have interaction.

2.2 Latent Factor Models

Latent factor models map both users and items onto a joint D-dimensional latent space. We denote user latent vectors as $U = [\boldsymbol{u_1}, ..., \boldsymbol{u_N}]$ and item latent vectors as $V = [\boldsymbol{v_1}, ..., \boldsymbol{v_M}]$, where $D \ll \min(M, N)$. The preference score $\hat{R}_{ij}$ is estimated as :

$$\hat{R}_{ij} = \left\langle \boldsymbol{u_i}, \boldsymbol{v_j} \right\rangle = \boldsymbol{u_i}^T \boldsymbol{v_j}, \tag{1}$$

where $\boldsymbol{u_i}$ is the latent factor of the i-th user and $\boldsymbol{v_j}$ is the latent factor of the j-th item.

The objective is to minimize the following regularized squared loss on observed ratings:

$$\arg\min_{\mathrm{U,V}} \sum_{(i,j) \in \mathcal{R}} \left(R_{ij} - \hat{R}_{ij} \right)^2 + \lambda \cdot \left(\|U\|^2 + \|V\|^2 \right), \tag{2}$$

where λ is the regularization parameter which is usually an L_2 norm to prevent overfitting. We then calculate the preference scores by using inner product of the two latent factor vectors and rank the items according to the estimated preference score $\hat{R}_{ij}$.

2.3 Bayesian Personalized Ranking

BPR model is a framework for addressing the implicitness in CF [13]. BPR models a triplet of one user and two items, where one of the items receives positive feedback while the other does not. The widely used BPR objective is given as :

$$\arg\min_{\mathrm{U,V}} \sum_{(i,j,k) \in \mathcal{R}_{\mathcal{B}}} -\ln \sigma \left(\hat{R}_{ij} - \hat{R}_{ik} \right) + \lambda \cdot \left(\|U\|^2 + \|V\|^2 \right), \tag{3}$$

where σ is the logistic sigmoid function. The training data $\mathcal{R}_{\mathcal{B}}$ is generated as:

$$\mathcal{R}_{\mathcal{B}} = \{(i, j, k) | j \in \mathcal{R}(i) \wedge j \in \mathcal{I} \setminus \mathcal{R}(i)\}, \tag{4}$$

where $\mathcal{I}$ denotes the set of all items in the dataset and $\mathcal{R}(i)$ represents the set of items that are interacted by the i-th user. The semantics of $(i, j, k) \in \mathcal{R}_{\mathcal{B}}$ is that user i is assumed to prefer item j over k.

In this work, we use BPR as our basic learning model.

2.4 Graph Convolutional Networks

Graphs provide a universal language to represent relationships between entities of the same type as well as of different types such as the user and the item [5]. If we see users and items as nodes of a bipartite graph, recommendation can be seen as a problem of link prediction. Graph Convolutional Networks have proved effective in various tasks related to graphs [7, 8], including collaborative filtering recommendation [17], but there is not much attention paid on content-aware recommendation.

Graph Convolutional Networks take adjacency matrix A and feature matrix X as the input, and produce the node-level output Z. Layer-wise propagation rule is given as :

$$H^{(l+1)} = f\left(H^{(l)}, A\right) = a\left(A \cdot H^{(l)} \cdot W^{(l)}\right), \tag{5}$$

where $W^{(l)}$ is a weight matrix for the l-th neural network layer and a is a non-linear activation function like sigmoid or ReLU.

However, we can not directly use Equation (5) for our task. When we apply it to recommendation, the adjacency matrix is not symmetric anymore, since the number of user doesn't equal to that of item.

3 CONTENT-AWARE RECOMMENDATION WITH GRAPH AUTOENCODER

In this section, we will introduce our Graph Content-aware Recommendation (GraphCAR) model in detail. First, we present the general GraphCAR framework, elaborating the motivation of the model. We then show the formulations of the proposed encoder and decoder. Lastly, we will go through the optimization of GraphCAR.

3.1 General Framework

GraphCAR is a neural network that models user's preference scores with respect to the multimedia content. We use two-layer graph convolutional networks to encode users and items, respectively. Then we use the inner product of the two latent factor vectors to estimate preference scores. We calculate the loss and use back-propagation to update parameters in graph convolutional networks. After we obtain the optimized user and item latent factor vectors, recommendation is then simplified to a ranking problem according to the estimated scores.

3.2 Encoder: Graph Convolutional Networks

Suppose there are m users and n items. Each user has attributes of d_u-dimension and each item has a feature of d_i-dimension. The goal of the encoder is to generate latent factors of all users and items. We employ graph convolutional networks to encode. We propose a novel two-layer **asymmetric** graph convolutional network to compute the latent representation of users and items. Given the user-item interaction matrix R (whose size is $m \times n$) and the user

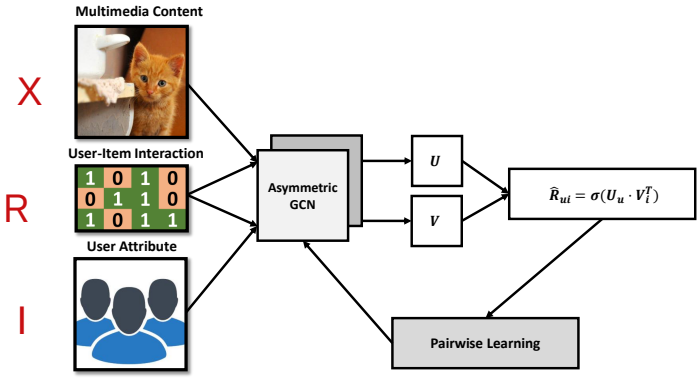

Figure 1: The architecture of our proposed content-aware recommendation with graph autoencoder.

attributes matrix I (whose size is $m \times d_u$) and the item feature matrix X (whose size is $n \times d_i$), the latent factor of users is given as :

这里有没有通过对R的两层GCN完成user和item的信息交互？

$$U_0 = ReLU\left(R^T \cdot I \cdot P_0\right), \tag{6}$$

$$U = \sigma\left(R \cdot U_0 \cdot P_1\right), \tag{7}$$

where σ is the logistic sigmoid function and P_0, P_1 are two weight matrices and .

Similar to Equation (6, 7), the latent factor of items is given as:

$$V_0 = ReLU\left(R \cdot X \cdot W_0\right), \tag{8}$$

$$V = \sigma\left(R^T \cdot V_0 \cdot W_1\right). \tag{9}$$

3.3 Decoder and Objective Function

Generated from Equation (6, 7 ,8, 9), latent factors U and V have the same column number$d$. We follow the method of LFM (Equation (1)) to calculate preference score :

$$\hat{R} = U \cdot V^T. \tag{10}$$

Object Function. GraphCAR is optimized in the BPR pairwise learning objective [13]: optimizing the pairwise ranking between the positive and non-observable items:

$$\arg\min_{\Theta} \sum_{(u,i,j)\in\mathcal{R}_B} -\ln\sigma\hat{R}_{uij} + \frac{\lambda}{2}\cdot\left(\sum_{\Theta\in\{W_0,W_1,P_0,P_1\}}\|\Theta\|^2\right), \tag{11}$$

where σ is the logistic sigmoid function and λ is regularization parameter. Due to space limit, we use Θ to denote the set of parameters in graph convolutional networks and use $\hat{R}_{uij}$ to denote $\hat{R}_{ui} - \hat{R}_{uj}$.

3.4 Algorithm

A stochastic gradient descent algorithm based on a bootstrap sampling of training triplets is proposed to solve the network.

For notational simplicity, we divide GraphCAR into 5 steps: 1) We calculate preference scores of all users and items. 2) We sample a mini-batch from all users and generate training triplets of users of this batch. 3) Calculate the BPR loss of this batch and update parameters of GraphCAR according to backpropagation. 4) Generate next batch and repeat 2-3. 5) Repeat step 1-4 until the objective function convergence.

4 EXPERIMENTS

In this section, we will conduct experiments to answer the following research questions:

- **RQ1** Does GraphCAR outperform state-of-the-art recommendation methods?
- **RQ2** Aimed at users of different sparsity levels, how the performance of the GraphCAR?

We will first present the experiment settings, followed by answering the above two research questions, and end with some illustrative examples.

4.1 Experimental Settings

Datasets. We experimented with two publicly accessible datasets: Amazon Movies and TVs (Amazon)[1] and Vine[2]. The characteristics of the two datasets are summarized in Table 1. The Amazon Movies and TVs dataset is constructed by [4, 11]. Due to the high sparsity of the dataset, we filter the dataset by retaining the users who have rated at least fifty movies. The Vine dataset is used in [2]. We filter the dataset by retaining users with at least 10 interactions.

Table 1: Statistics of the evaluation datasets.

Dataset	#Interaction	#Item	#User	Sparsity
Amazon	606, 033	95, 502	4, 319	99.85%
Vine	330, 793	273, 352	5, 016	99.98%

Evaluation Protocols. We adopt the hold-one-out evaluation method which has been used in [13]. For each user, we leave his/her latest interaction as the test set and take the remaining data for training. Based on the ranked preference scores, we adopt Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) [5] to measure whether the ground-truth item is present on the ranked list and where the position of it is. If not specifically specified, we truncate the ranked list at 100 among all the items for both metrics.

Baselines. We compare GraphCAR with the following algorithms: ItemKNN [14], BPR [13], SVD++ [9], and ACF [2]. Note that all model-based CF models are learned by optimizing the same pairwise ranking loss of BPR for a fair comparison.

Feature Extraction. Amazon dataset provides a 4096-dimensional visual feature of each item, which is extracted by using a deep CNN network [11]. For Vine, we adopted the widely-used architecture ResNet-50 [3] to extract visual features for frames of videos.

Parameter Settings. For models that are based on MF, we randomly initialize parameters with a Gaussian distribution ($\mu = 0, \sigma = 0.01$), and optimize the model with stochastic gradient descent (SGD). We test the batch size of [128, 256, 512], the latent feature dimension of [8, 16, 32, 64, 128], the learning rate of [0.001, 0.003,

[1] http://jmcauley.ucsd.edu/data/amazon
[2] https://drive.google.com/file/d/0B_nM-g91nfPZa3Y4YUZTUWFxalU/view

Table 2: Performance of HR@100 and NDCG@100.

Method	Amazon		Vine	
	HR	NDCG	HR	NDCG
ItemKNN	0.317	0.0679	0.621	0.1692
BPR	0.330	0.0814	0.629	0.1763
SVD++	0.322	0.0845	0.645	0.1796
ACF	0.341	0.0911	0.667	**0.1908**
GraphCAR	**0.352**	**0.0938**	**0.670**	0.1903

0.01, 0.03, 0.1] and regularizer of [0, 0.0001, 0.0003, 0.001, 0.003, 0.01, 0.1]. We only show the results of $D = 128$, a relatively large number that returns good accuracy.

4.2 Model Comparison (RQ1)

Table 2 shows the performance of HR@100 and NDCG@100 of each method of the best configuration. From the table, we can observe that our proposed method achieves the best performance of HR on both datasets, significantly outperforming the state-of-the-art MF and Hybrid methods. While on the Vine dataset our performance of NDCG is slightly beaten by that of ACF, our method performs better on the Amazon dataset.

4.3 Model Analysis: Performance over Users of Different Sparsity Levels (RQ2)

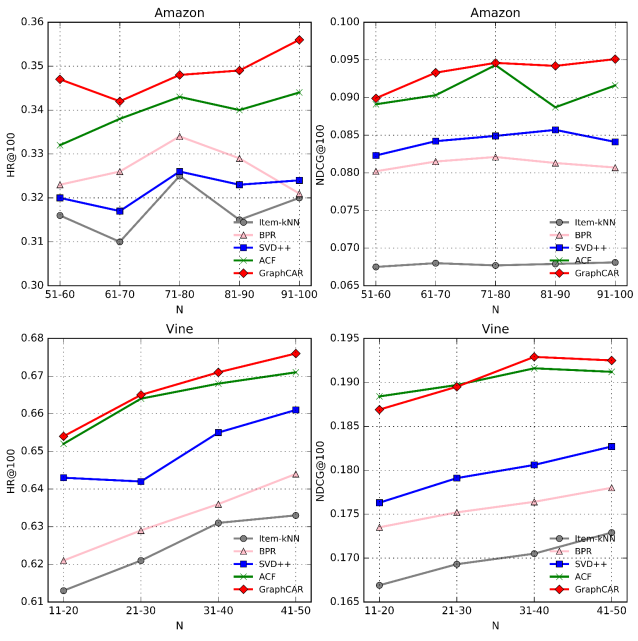

Figure 2: Performance of HR@100 and NDCG@100 w.r.t. the number of items per user on two datasets.

To investigate the performance of our model over users of different sparsity levels, we show the performance with respect to the number of items a user has interacted with. From Figure 2, we observed that when users have fewer interactions with items, our method performs good, while the advantages are not that distinct. Our proposed method shows great advantages for users who have interactions with items densely. Under this circumstance, our model significantly outperform competing methods.

5 CONCLUSIONS

In this work, we propose a novel content-aware recommendation model with graph autoencoder, to address the implicit feedback in multimedia and combine CF method with the multimedia content to achieve better performance. To the best of our knowledge, GraphCAR is the first work that exploits graph convolutional networks to content-aware recommendation. We conducted experiments on two datasets and demonstrated that GraphCAR can outperform the state-of-the-art models in multimedia recommendation. In future, we plan to extend GraphCAR in discrete models [10, 15, 16, 18] to reach higher efficiency. Moreover, we would explore a more complicated decoder such as NCF [6].

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Project 61502081 and Project 61632007, in part by the Fundamental Research Funds for the Central Universities under Project ZYGX2014Z007.

REFERENCES

[1] Deepak Agarwal and Bee-Chung Chen. 2009. Regression-based latent factor models. In *Proc. KDD*. 19–28.
[2] Jingyuan Chen, Hanwang Zhang, Xiangnan He, Liqiang Nie, Wei Liu, and Tat-Seng Chua. 2017. Attentive Collaborative Filtering: Multimedia Recommendation with Item- and Component-Level Attention. In *Proc. SIGIR*. 335–344.
[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. CVPR*. 770–778.
[4] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proc. WWW*. 507–517.
[5] Xiangnan He, Ming Gao, Min-Yen Kan, and Dingxian Wang. 2017. BiRank: Towards Ranking on Bipartite Graphs. *IEEE TKDE* 29, 1 (2017), 57–71.
[6] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proc. WWW*. 173–182.
[7] Thomas N. Kipf and Max Welling. 2016. Semi-Supervised Classification with Graph Convolutional Networks. *CoRR* abs/1609.02907 (2016).
[8] Thomas N. Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. *CoRR* abs/1611.07308 (2016).
[9] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proc. KDD*. 426–434.
[10] Defu Lian, Rui Liu, Yong Ge, Kai Zheng, Xing Xie, and Longbing Cao. 2017. Discrete Content-aware Matrix Factorization. In *Proc. KDD*. 325–334.
[11] Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-Based Recommendations on Styles and Substitutes. In *Proc. SIGIR*. 43–52.
[12] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM TIST* 3, 3 (2012), 57:1–57:22.
[13] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proc. UAI*. 452–461.
[14] Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proc. WWW*. 285–295.
[15] Fumin Shen, Yadong Mu, Yang Yang, Wei Liu, Li Liu, Jingkuan Song, and Heng Tao Shen. 2017. Classification by Retrieval: Binarizing Data and Classifiers. In *Proc. SIGIR*. 595–604.
[16] Fumin Shen, Chunhua Shen, Wei Liu, and Heng Tao Shen. 2015. Supervised Discrete Hashing. In *Proc. CVPR*. 37–45.
[17] Rianne van den Berg, Thomas N. Kipf, and Max Welling. 2017. Graph Convolutional Matrix Completion. *CoRR* abs/1706.02263 (2017).
[18] Hanwang Zhang, Fumin Shen, Wei Liu, Xiangnan He, Huanbo Luan, and Tat-Seng Chua. 2016. Discrete Collaborative Filtering. In *Proc. SIGIR*. 325–334.