Socially Enhanced Spatial Temporal Knowledge Graph Convolutional Network(SESTKGCN)

Project-II (CS47006) report submitted to

Indian Institute of Technology, Kharagpur

in partial fulfilment for the award of the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

Jyoti Agrawal

(17CS10016)

Under the supervision of

Prof. Pabitra Mitra

Department of Computer Science and Engineering

Indian Institute of Technology, Kharagpur

Autumn Semester, 2016-17

May 04, 2021

DECLARATION

I certify that

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Date: May 04, 2021 (Jyoti Agrawal)
Place: Kharagpur (17CS10016)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KHARAGPUR

KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Socially Enhanced Spatial Temporal Knowledge Graph Convolutional Network(SESTKGCN)**" submitted by **Jyoti Agrawal** (Roll No. 17CS10016) to Indian Institute of Technology, Kharagpur towards partial fulfilment of requirements for the award of degree of Bachelor of Technology in Computer Science and Engineering is a record of bona fide work carried out by her under my supervision and guidance during Autumn Semester, 2016-17.

Pabitra Mitra

Prof. Pabitra Mitra
Department of Computer Science and Engineering
Indian Institute of Technology,
Kharagpur
Kharagpur - 721302, India

Date: May 04, 2021

Place: Kharagpur

Abstract

Name of the student: **Jyoti Agrawal** Roll No: **17CS10016**

Degree for which submitted: **Bachelor of Technology**

Department: **Department of Computer Science and Engineering**

Thesis title: **Socially Enhanced Spatial Temporal Knowledge Graph Convolutional Network(SESTKGCN)**

Thesis supervisor: **Prof. Pabitra Mitra**

Month and year of thesis submission: **May 04, 2021**

Item recommendation is the task of predicting a personalized ranking on a set of items(e.g. websites, movies, products). We need to take side information that we get from different sources about item and user into account to provide more accurate, diverse, and explainable recommendation. Traditional methods like factorization machine(FM) make it a supervised learning problem, and assumes each interaction as an independent instance with side information encoded. Due to the overlook of the relations among items (e.g., the author of a book is also author of another book), these methods become insufficient to extract the collaborative signal from the collective behaviors of users.

This algorithm provides a very generic view of GCN Based Recommendation systems without the need of any input input features on item or user. Here we incorporate both, item Knowledge graph and User social media network with user-item bipartite graph. This algorithm basically works on the merge of all the three above mentioned

graphs (user-user graph, user-item bipartite graph, item Knowledge graph). It generates independent embeddings for user and item believing that the properties of item does not depend on users and similarly the properties, a user is interested in does not depend on the items it purchased and thus the embeddings of any item must not change with every user and vice versa. But it definitely believes that a property of an item is strong if it has more users interested in that property. Similarly the interest of a user in a particular property is strong if it purchases more items with that property as dominating property and this information will be used to aggregate users and items embeddings from neighbours.

This algorithm also tries to incorporate the position the user is situated at and the time of purchase of the item using the concept of Spatial GCN.

Acknowledgements

I would like to thank my supervisor Dr. Pabitra Mitra, Associate Professor, Department of Computer Science and Engineering, IIT Kharagpur for his valuable guidance and support all throughout the course of my B.Tech project work.

I would like to thank Bithika Pal, PhD Scholar, department of Computer Science and Engineering, IIT Kharagpur, who have guided throughout this project.

I would like to express my gratitude to all faculty members of Department of Computer Science and Engineering for guiding us and providing knowledge in various subjects over the past three years. Finally, my deep and sincere gratitude to my family and friends in IIT Kharagpur for their continuous love and support

Contents

List of Figures

List of Tables

Abbreviations

SESTKGCN	**S**ocially **E**nhanced **S**patial **T**emporal **K**nowledge **G**raph **C**onvolutional **N**etwork
KGCN	**K**nowledge **G**raph **C**onvolutional **N**etwork
KG	**K**nowledge **G**raph
SG	**S**ocial **G**raph

Chapter 1

Introduction

Recommender system helps user discover items they like. Today with the internet reaching nearly everyone, recommendation system has become a very important and integral part of trade. A lot of trade has started online. Examples are, e-commerce websites, online movie sites, news, real estate websites, tourism websites, etc.

You can see messages like: "frequently bought together", "recommended for you", "because you bought this", etc on these business websites. This provides you with personalised experience to attract you towards a product you may like.

With the rapid development of e-commerce and social media platforms, we have got so much data on user-user interaction, user-item interaction and item Knowledge graph.

Both the user interests and the item properties can be represented with compressed vectors. Hence, to learn user/item embeddings with historical interactions and other side information such as social relationship and knowledge graph is the main challenge in this field. In recommender systems, most information has graph structure. For example, social relationship among users and knowledge graph related to items

are naturally graph data. In addition, the interactions between users and items can be considered as the bipartite graph, and the item transitions in sequences can be constructed as graphs as well. Therefore, graph learning approaches have been leveraged to get user/item embeddings.

This algorithm uses GCN based approach to learn embeddings of both user and item from the input feature vectors (either available or learned). Assume a combined graph of the three graphs, i.e., user social network, user-item bipartite graph and item Knowledge graph say G. You have input feature vectors of each entity node(user or item) and also input vectors corresponding to relationships in item Knowledge.

As this algorithm is trying to capture a very generic image, we try to assume we have as much information as possible. You can omit parts as per your data set. This approach assumes that we know when an item was rated/purchased. We also know the rating given to an item by a user and the number of votes given to that rating. We even have the information of position of users. We also have strength of every user to user friendship. We would also have item-item relationship information. This algorithm combines all this information to generate embeddings for every user and item. We call the combined graph of the three graphs as Combined user-item graph.

This algorithm also uses concept of Spatial Graph Convolutional(Spatial GCN) to take care of the position of the users known. It uses temporal GCN to take care of the time information for user-item interaction. This would run GCN with some weights while aggregating, which would be explained in detail in the algorithm.

The basic concept of this algorithm is same as matrix factorization, i.e, consider you have user-item interaction matrix(M_{u*v}) which contains the rating provided to each item by a user. Now you can consider this as a product of two matrices of I_{u*e} and transpose of F_{v*e}. So you can think of matrix I as a matrix of embeddings of user's interest in a feature and matrix F as the matrix of feature vectors corresponding to

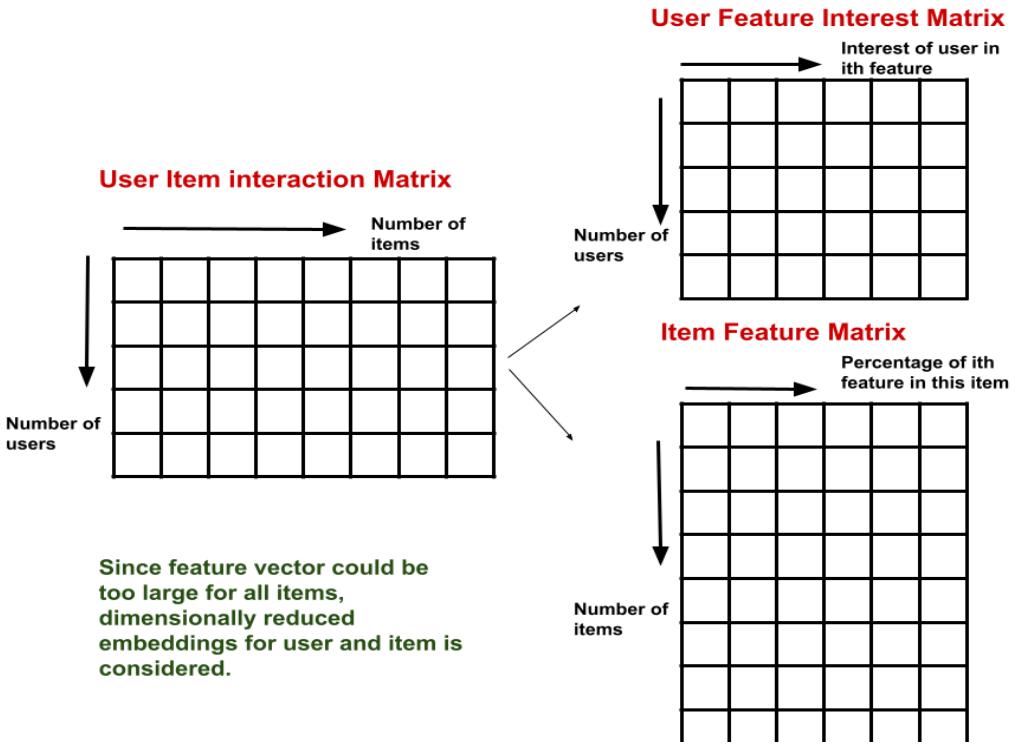


FIGURE 1.1: Matrix Factorization Concept.

each item. So we consider the product of user embedding and item embedding as the predicted rating for that pair. Since feature vector for a particular item can be too large, we consider reduced dimension embedding instead of direct feature vectors.

This algorithm is also based on learning of these user and item embeddings and getting the probability of a particular user being interested in a particular item as the dot product of these embedding vectors.

Our contribution in this paper are summarized as follows:

- We propose the end-to-end model for recommendation incorporating the item Knowledge Graph, User Social Network Graph and the User Item interaction bipartite graph.
- We show the results of out model on three real life recommendation scenarios, i.e., movie, music and books.

- We release the code of SESTKGCN and the datasets at `https://github.com/jyoti246/SESTKGCN` for validation and future work.

Chapter 2

Related Work

2.1 Motivation

There has been a paradigm shift in mainstream modeling of recommender systems in the past decades, evolving from neighbourhood methods to frameworks based on representation learning. Recommendations by item-based neighborhood methods are computed using the similarity of the historical data items shared by the user. In a sense, they represent the preferences of the user by directly computing with their historical interacted items. Because of their simplicity, efficiency, and effectiveness, these early item-based neighbourhood approaches have been rewarded with phenomenal success in real-world applications. There is an alternative approach that tries to encode both items and users as embeddings in a common space, thus enabling us to compare them directly. These are the foundations for the representation learning based methods. The Netflix Prize competition has demonstrated that the matrix factorization models are better compared to classic neighborhood methods for the task of recommendations. Since then, there has been a huge surge of interest in representation-based models. Starting from matrix factorization to deep learning

models, there are numerous proposals for various methods to learn users and items representation for better estimate/prediction of the preferences of users on items. As of today, DL models have been a prevailing philosophy for recommender frameworks in both scholarly exploration and mechanical applications because of the capacity in viably catching the non-linear and non-trivial user-item relationships and effectively fusing plentiful information sources, e.g., textual, visual and contextual information.

Among every one of those DL algorithms, Graph Nueral Network (GNN) is without a doubt and highly alluring method as a result of its predominant capacity in learning on information of graph-based, which is important for any recommender frameworks. For example, we can represent the interaction data in a recommendation application through a bipartite graph between item nodes and user nodes, and the interactions which are observed represented by edges. The transition of items in an user behavioral sequences may also be modelled as graphs. The advantage of such formulation of recommendation tasks on graphs turns out to be particularly obvious when amalgamating structured external information, e.g., knowledge graph related to items and the social relationship among users. Along these lines, GNN gives an unified point of view to demonstrate the plentiful heterogeneous information in recommender frameworks.

With these benefits, GNN has made noteworthy progress in recommender frameworks in the previous few years. In scholarly examination, a great deal of works exhibit that GNN-based models beat past strategies and accomplish new cutting edge results on the public benchmark datasets. In the interim, a lot of their variations are suggested and applied to different suggestion undertakings, e.g., Points-of-interest (POI) recommendation, session based recommendation, bundle recommendation and group recommendation. In industry, GNN has additionally been deployed in large web-scale recommender frameworks to create excellent recommendation results. For instance, random walk-based Graph Convolution Network (GCN)

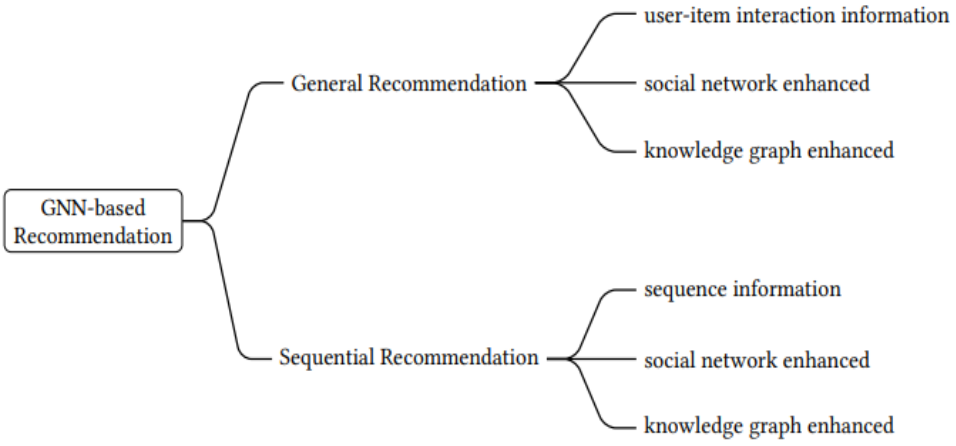


FIGURE 2.1: Categories of graph neural network based recommendation models.

calculation model named PinSage, has been created and deployed by Pinterest on a graph with 3 billion nodes and 18 billion edges, and acquired generous enhancements in client commitment in an online A/B test.

2.2 Categorization

Recommendation Systems can be divided in two categories as follows.

2.2.1 General recommendation

The assumption behind general recommendation is that the users preferences are static and models them based on either implicit (e.g., clicks, reads, or purchases) or explicit (i.e., ratings) feedback. A common approach for this type of model is to reconstruct historical interactions of users' by the users and items representations.

Having item set I, the user set U, and the observed interactions between users and items $R : UI$, for any user $u \in U$ general recommendation is to estimate her/his

preference for any item $i \in I$ by the representation learnt for user h_u^* and item representation h_i^* , i.e.,

$$y_{u,i} = f(h_u^*, h_i^*)$$

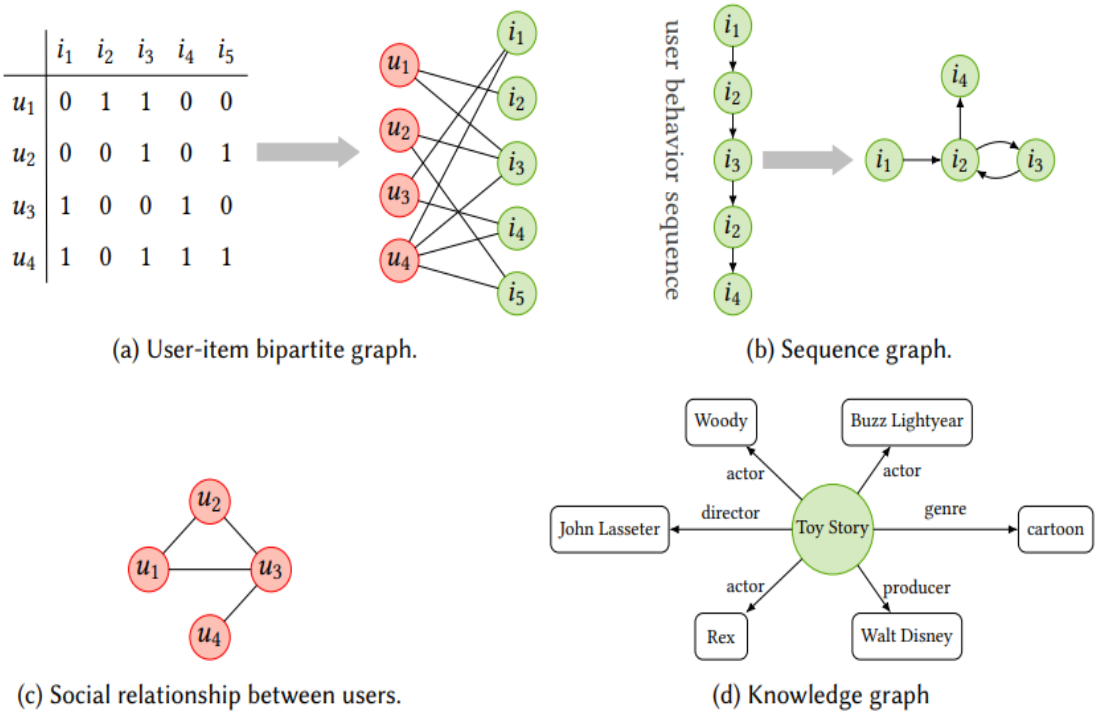
where score function $f(.)$ can be cosine, multi-layer perceptions, dot product, etc., and the preference score for user u on item i, denoted by $y_{u,i}$ is usually presented in probability.

There exist bountiful works regarding the general recommendation task. Most early research considers using user-item interactions in the form of the matrix, formulating the recommendation as the matrix completion task. Hereby, Matrix Factorization (MF) projects users and items into a shared vector space and subsequently reconstructs the whole user-item interaction matrix, i.e., predicting an users' preferences on their unseen item. Presently, recommender systems have been significantly revolutionized by deep learning.. One line of such research tries to improve the recommendation performance by incorporating helper data such as text and images, with the power of deep learning.

2.2.2 Sequential recommendation

Complementing general recommendation, there is another mainstream research direction in recommender systems, which takes into account the dynamic and evolving nature of user's preference. Sequential recommendation seeks to predict the successive item(s) that a user is likely to interact with by investigating the sequential patterns in her/his historical interactions.

In sequential recommendation, users' historical interactions are organized into sequences in a chronological order. We represent the interaction sequence for user u



(a) User-item bipartite graph. (b) Sequence graph.

(c) Social relationship between users. (d) Knowledge graph

FIGURE 2.2: Representative graph structures in recommender systems.

with $s^u = [i_{s,1}, i_{s,2}, ..., i_{s,n}]$, where $i_{s,t}$ denotes the item which user u interacts at the time step t and n denotes the length of the interaction sequence. The sequential recommendation task is to predict the most possible item which the user u will interact with at the next time step $n + 1$.

2.3 General Recommendation Systems

General recommendation can again be categorised as follows based on the amount of information available.

Information	Graph	Model	Venue	Year	GNN Framework
user-item interactions	Bi[1]	GC-MC [104]	KDD	2018	variant of GCN (mean-pooling)
	Bi	PinSage [145]	KDD	2018	variant of GraphSage (importance-pooling)
	Bi	SpectralCF [159]	RecSys	2018	variant of GCN
	Bi	STAR-GCN [151]	IJCAI	2019	variant of GCN (mean-pooling)
	Bi	NGCF [119]	SIGIR	2019	variant of GraphSage (node affinity)
	Bi	Bi-HGNN [55]	IJCAI	2019	GraphSage
	Bi	LR-GCCF [10]	AAAI	2020	variant of GCN (w/o activation)
	Bi	LightGCN [31]	arxiv	2020	variant of GCN (w/o activation & transformation)
	Bi	IG-MC [152]	ICLR	2020	variant of GraphSage (sum updater)
	Bi	DGCN-BinCF [109]	IJCAI	2019	variant of GCN (CrossNet)
	Bi	HashGNN [101]	WWW	2020	variant of GCN (mean-pooling)
	Bi	NIA-GCN [97]	SIGIR	2020	variant of GraphSage (neighbor interaction)
	Bi	AGCN [134]	SIGIR	2020	variant of GCN (w/o activation)
	Bi	MBGCN [42]	SIGIR	2020	hiearchical aggregation
	Bi	MCCF [123]	AAAI	2020	GAT
	Bi	DGCF [121]	SIGIR	2020	hiearchical aggregation
	Bi	GraphSAIL [144]	CIKM	2020	GraphSage
	Bi	DisenHAN [125]	CIKM	2020	hiearchical aggregation
	Bi	Gemini [143]	KDD	2020	variant of GAT
	Bi	Multi-GCCF [98]	ICDM	2019	GraphSage
	Bi	DGCF [72]	arxiv	2020	variant of GCN (w/o activation & transformation)
	Bi	TransGRec [133]	SIGIR	2020	variant of GraphSage (sum updater)
user-item interactions & social network (SN)	SN	DiffNet [132]	SIGIR	2019	GraphSage
	Bi + SN	GraphRec [17]	WWW	2019	variant of GAT
	SN + I2I	DANSER [135]	WWW	2019	variant of GAT
	Bi + SN	DiffNet++ [131]	TKDE	2020	variant of GAT
	Motif	ESRF [147]	TKDE	2020	mean-pooling
user-item interactions & knowledge graph (KG)	KG	KGCN [114]	WWW	2019	variant of GAT
	KG	KGNN-LS [112]	KDD	2019	variant of GCN (user-specific adjacent matrix)
	Bi + KG	KGAT [118]	KDD	2019	variant of GAT
	U2U + I2I	IntentGC [158]	KDD	2019	variant of GraphSage (sum updater)
	KG	AKGE [91]	arxiv	2019	variant of GAT
	Bi + KG	MKGAT [99]	CIKM	2020	variant of GAT
	Bi + KG	ACKRec [22]	SIGIR	2020	meta-path based
	KG	ATBRG [18]	SIGIR	2020	variant of GAT
	Bi + KG	TGCN [8]	CIKM	2020	variant of GAT (hierarchical)

[1] Bi represents bipartite graph.

FIGURE 2.3: Graph neural network based general recommendation.

2.3.1 User-item interaction information

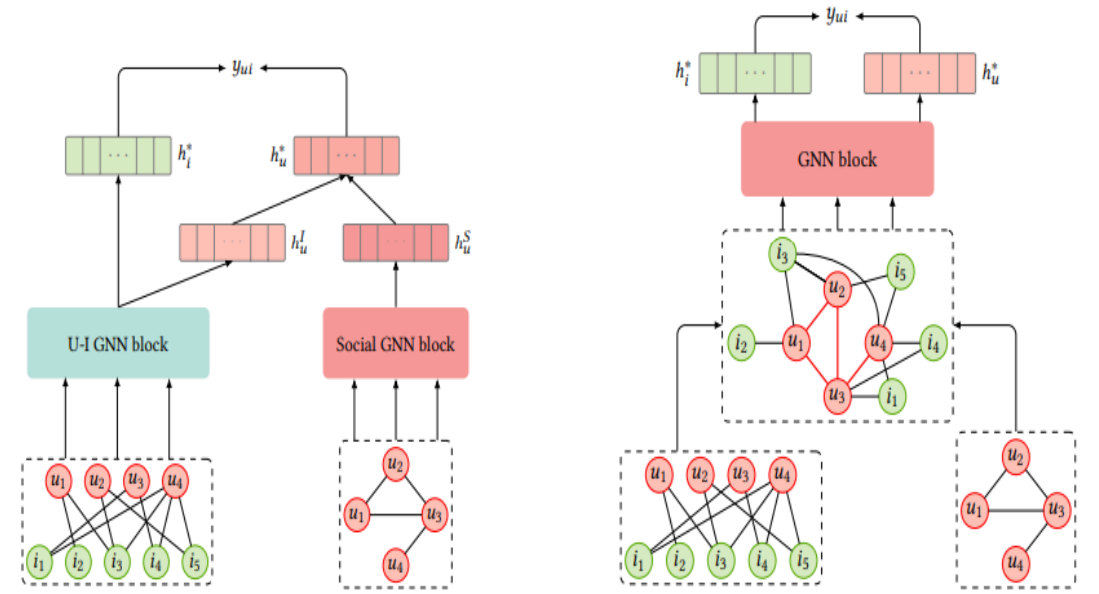
The fundamental idea is essentially to improve user representation using the items interacted by users and using the items interacted atleast once by the user to enhance representation of the item. In this way, information diffusion processes can be simulated by multi-layer GNN methods and higher-order connectivity from user-item interactions can be exploited more efficiently. Assuming we have the user-item bipartite graph, the main challenge is propagating the information of interacted items/users to the user/item, and learning the final user/item representations for the task of prediction.

2.3.2 Social network enhanced

With the rise of online informal communities, social recommender frameworks have been proposed to use every user's local neighbors' preferences for enhancing user modeling. The basic assumption behind all these works is that users with social relationships should have similar representations based on the social influence hypothesis that associated individuals would impact one another.Some approaches directly use such relationships as regularizers to constraint the final user embeddings, while others leverage such relationships as input to influence the original user embeddings .

2.3.3 Knowledge graph enhanced

User representation is enhanced by utilizing social networks that reflect relationships between users, while item representation is enhanced by leveraging a knowledge graph that expresses relationships between items through attributes. The incorporation of knowledge graph into recommendation offers two benefits : (1) the rich



(a) The framework of GNN on the bipartite graph and social network graph separately.

(b) The framework of GNN on the unified graph of user-item interactions and social network.

FIGURE 2.4: Two strategies for social enhanced general recommendation

semantic similarity among items in a knowledge graph helps in exploring their connections and thus enriching item representation (2) knowledge graph connects the historically interacted user's items and recommended items, which aids the interpretability of the results. However, making use of knowledge graph in recommendation is quite challenging due to its structural complexity, i.e., multi-type entities and multi-type relations.

Chapter 3

Task Formulation

As we need to run a GCN on one combined graph of all the Information we have, we process our data and get it in required form as mentioned below.

3.1 KG Preprocessing

Before moving further, Knowledge Graph needs to be constructed as per the requirements of this algorithm.

The requirement is that the Knowledge Graph must contain nodes of only item type with edges showing type of relationship between them. No other type of node must be present.

To gain the KG as per above condition you need to reduce every node, other then item node, to an edge. Here is an Example shown for a Knowledge Graph for items of type movie.

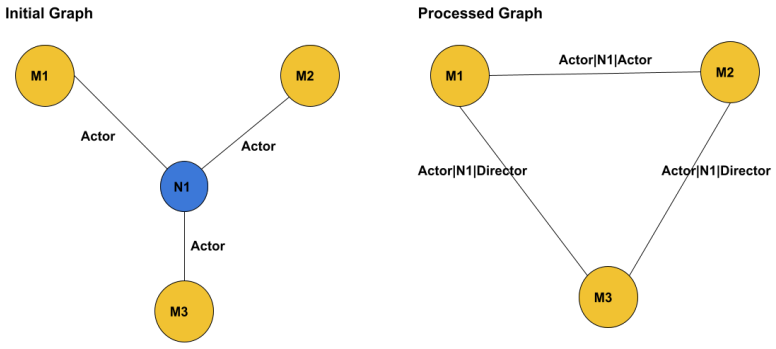


FIGURE 3.1: Input Knowledge Graph and the form of Graph we need.

3.2 Combined User-Item Graph

Let us now prepare the final graph, we want to learn our algorithm on, by combining the three graphs, namely item-KG, User-Item Bipartite Graph and user social networking graph. You can simply ignore social Network graph or the item KG if you do not have the information in your data.

Combined graph is just an understanding where we have item-item interaction which comes from item-KG, we have user-item interaction which comes from User-Item Bipartite Graph and we have user-user interaction which comes from social network graph.

3.3 Problem Formulation

We formulate the socially enhanced knowledge graph based recommendation problem as follows. In a typical recommendation scenario, we have a set of M users U = u_1,u_2, ...,u_m and a set of N items V = v_1,v_2, ...,v_n

We have the user item interaction information which is based of users feedback to a particular item. If a user has given positive feedback to an item then the its 1 else

its 0. Additionally we have the User-Item Graph as a combination of the 3 graphs as explained above.

Given the user-item interaction matrix Y, the user position information, as well as the User-Item G, we aim to predict whether user u has potential interest in item v with which he has had no interaction before. Our goal is to learn a prediction function $y_{uv} = F(u, v|\theta, Y, G)$, where y_{uv} denotes the probability that user u will engage with item v, and θ denotes the model parameters of function F.

Chapter 4

Methodology

4.1 SESTKGCN layer

The algorithm runs through layers. In the first layer we have as inputs, the feature vector corresponding to every user, item and item-item relationship denoted by $h_u^{(0)}$, $h_i^{(0)}$ and $h_r^{(0)}$ respectively. We also know position of each user (p_u), strength of user-user friendship between u and u' ($s_u^{u'}$), rating given by user u to item i (rt_i^u) and the number of votes given to this rating (v_i^u) and time (t_i^u) when item i was rated by user u.

We need to update embedding of node of type user, item and the embedding of each kind of relationships at each layer. As there are three different type of updates we would explain each update separately.

4.1.1 Update Item node Embedding

The node i of type item will take the aggregate of all the neighbouring User type nodes and all the neighbouring Item type nodes separately as $h_{N^U(i)}^{(k)}$ and $h_{N^I(i)}^{(k)}$

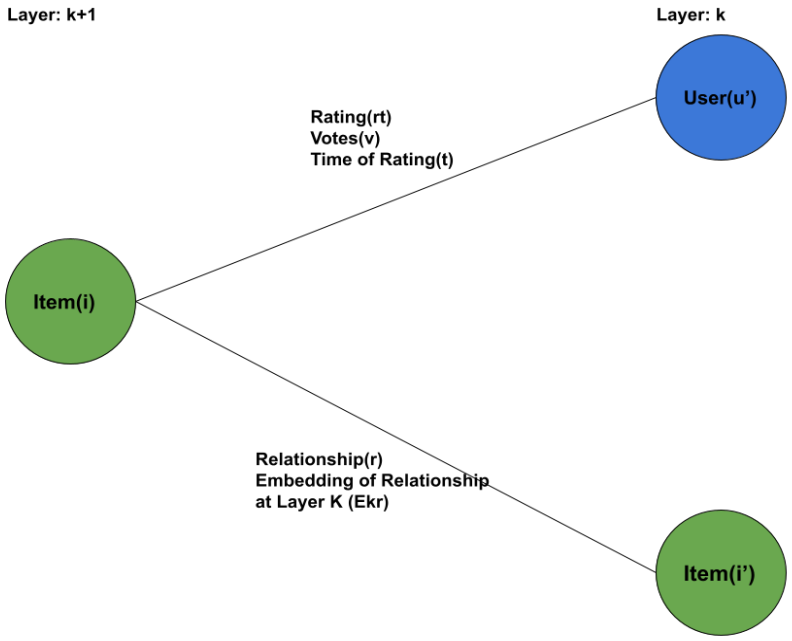


FIGURE 4.1: Update Item embedding using nodes of type User and Item separately.

respectively. Using these aggregates, the updated embedding $h_i^{(k+1)}$ of item i at (k+1)th layer using concatenation or sum aggregation respectively, would be

$$h_i^{(k+1)} = \sigma(\mathbf{W^k}.CONCAT(h_i^{(k)}, h_{N^U(i)}^{(k)}, h_{N^I(i)}^{(k)}) + b)$$

OR

$$h_i^{(k+1)} = \sigma(\mathbf{W^k}.(h_i^{(k)} + h_{N^U(i)}^{(k)} + h_{N^I(i)}^{(k)}) + b)$$

where $h_i^{(k)}$ is the embedding of item i at current layer and

$$h_{N^U(i)}^{(k)} = \sum_{u' \in N^U(i)} rt_i^{u'} * v_i^{u'} * (EMB(t_i^{u'}) \odot h_{u'}^{(k)})$$

and

$$h_{N^I(i)}^{(k)} = \sum_{i' \in N^I(i)} \pi_{r_i^{i'}, NORM}^{i} * h_{i'}^{(k)}$$

where $W^k \in \mathbb{R}^{d' \times d}$ is the trainable weight matrix and $\sigma()$ is non linearity. $\odot$ denotes element wise multiplication.

Here $EMB(t_i^{u'})$ is the vector we get for each time stamp using the concept of spatial temporal GCN which would be equal to

$$EMB(t) = \sigma(U^k.t + b)$$

where $U^k \in \mathbb{R}^{d \times d_(t)}$ and σ is non linearity. The usefulness of this was felt to incorporate the information of time into the model. Suppose some feature of a product becomes important during a particular time period say during a festival or it may happen a certain product loses importance with time and so its embedding must not me aggregated with much importance. Suppose we have time information in form of second, minute, hour, day, month, year, etc, so we can treat it as dimension of the input time vector and get an output vector of size of user-item embeddings after passing it through $EMB()$ function which can be multiplied element wise to put a weight at each embedding feature of the one being aggregated.

Also

$$\pi^i_{r_i^{i'}} = g(i, r_i^{i'})$$

where g: $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$(e.g. inner product of $h_i^{(k)}$ and $h^{(k)}_{r_i^{i'}}$). And so

$$\pi^i_{r_i^{i'},NORM} = \frac{exp(\pi^i_{r_i^{i'}})}{\sum_{i" \in N^I(i)} exp(\pi^i_{r_i^{i"}})}$$

This $\pi^i_{r_i^{i'}}$ can be considered as putting a weight to each neighbouring item(i') embedding getting aggregated for current item(i). The higher the item i can extract from the relationship $r_i^{i'}$ the higher would be the weight for aggregating the embedding of neighbour i'.

4.1.2 Update User node Embedding

The node u of type user will take the aggregate of all the neighbouring User type nodes and all the neighbouring Item type nodes separately as $h^{(k)}_{N^U(u)}$ and $h^{(k)}_{N^I(u)}$ respectively. Using these aggregates, the updated embedding $h^{(k+1)}_u$ of item u would be

$$h^{(k+1)}_u = \sigma(\mathbf{W^k}.CONCAT(h^{(k)}_u, h^{(k)}_{N^U(u)}, h^{(k)}_{N^I(u)}) + b)$$

OR

$$h^{(k+1)}_u = \sigma(\mathbf{W^k}.(h^{(k)}_u + h^{(k)}_{N^U(u)} + h^{(k)}_{N^I(u)}) + b)$$

where

$$h^{(k)}_{N^U(u)} = \sum_{u' \in N^U(u)} s^{u'}_u * (EMB'(pos^{u'}_u) \odot h^{(k)}_{u'})$$

and

$$h^{(k)}_{N^I(u)} = \sum_{i' \in N^I(u)} rt^u_{i'} * v^u_{i'} * (EMB(t^u_{i'}) \odot h^{(k)}_{i'})$$

where $W^k \in \mathbb{R}^{d' \times d}$ is the trainable weight matrix and $\sigma()$ is non linearity. $\odot$ is element wise multiplication.

Here $EMB'(pos^{u'}_u)$ is the vector we get for each relative position vector $pos^{u'}_u = |p_{u'} - p_u|$ using the concept of spatial GCN which would be equal to

$$EMB'(p) = \sigma(U'^k.p + b)$$

where $U'^k \in \mathbb{R}^{d \times x}$ where x is dimension of the position vector and σ is non linearity. This input position vector can be coordinates of position or some input embedding for position.

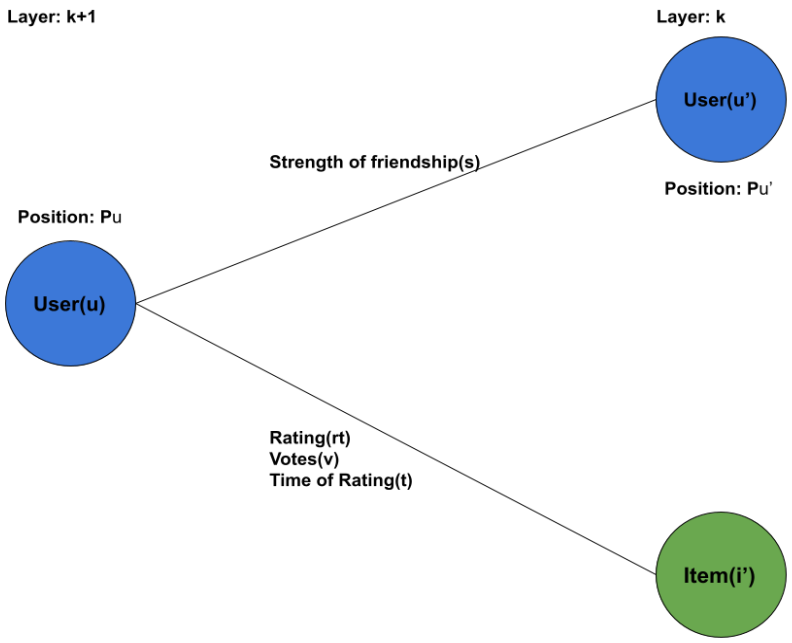


FIGURE 4.2: Update User embedding using nodes of type User and Item separately.

Also $EMB(t_i^{u'})$ is the vector we get for each time stamp using the concept of spatial GCN which would be equal to

$$EMB(t) = \sigma(U^k.t + b)$$

where $U^k \in \mathbb{R}^{d \times d_t}$ and σ is non linearity. Again where d_t denotes dimension of input time which would depend on how the input time is given.

4.1.3 Update Embedding of item-item Relationships

Embedding of each type of item-item relation for layer k+1, is

$$h_r^{(k+1)} = \sigma(\mathbf{W^k}.CONCAT(h_r^{(k)}, h_{L(r)}^{(k)}, h_{R(r)}^{(k)}) + b)$$

OR

$$h_r^{(k+1)} = \sigma(\mathbf{W^k}.(h_r^{(k)} + h_{L(r)}^{(k)} + h_{R(r)}^{(k)}) + b)$$

where L(r) and R(r) denotes left and right vertices corresponding to edges with relationship r. And

$$h_{L(r)}^{(k)} = AGGREGATE(h_i^{(k)}, FOR[i \in L(r)])$$

$$h_{R(r)}^{(k)} = AGGREGATE(h_i^{(k)}, FOR[i \in R(r)])$$

4.2 Learning the Parameters

The user, item embeddings we get as shown above is used to get the final probability score for any particular user(u) and item(v) pair. These embeddings are fed to a function $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ for predicting the probability:

$$y_{u,v} = f(h_u^K, h_v^K)$$

Algorithm 1: SESTKGCN algorithm

Input: Social graph SG(U,E); Item graph KG(V,R); User-Item interaction matrix Y; user set U; item set V; relationship set R; input features h_e^0, $\forall e \in (U \cup V \cup R)$; $weightmatrices \mathrm{W}_u^k$, W_v^k, W_r^k, U_{ut}^k, U_{vt}^k, U_{up}^k; Neighbourhood functions N_u^u, N_u^v, N_v^u, N_v^v; Relationships Left and Right Nodes L_r^v, R_r^v; Dot Product function f(u,v);

Output: User Embedding h_u^K, Item Embedding h_v^K

Data: Testing user u and item v

for $k = 1..K$ **do**
  for $u \in U$ **do**
    $h_{N^U(u)}^{(k-1)} \leftarrow agg(sample\ user\ neighbours)$
    $h_{N^I(u)}^{(k-1)} \leftarrow agg(sample\ item\ neighbours)$
    $h_u^k \leftarrow \sigma(W_u^k.COMBINE(h_u^{(k-1)}, h_{N^U(u)}^{(k-1)}, h_{N^I(u)}^{(k-1)}) + b)$
  for $v \in V$ **do**
    $h_{N^U(v)}^{(k-1)} \leftarrow agg(sample\ user\ neighbours)$
    $h_{N^I(v)}^{(k-1)} \leftarrow agg(sample\ item\ neighbours)$
    $h_v^k \leftarrow \sigma(W_v^k.COMBINE(h_v^{(k-1)}, h_{N^U(v)}^{(k-1)}, h_{N^I(v)}^{(k-1)}) + b)$
  for $r \in R$ **do**
    $h_{L(r)}^{(k-1)} \leftarrow agg(sample\ left\ item\ neighbours)$
    $h_{R(r)}^{(k-1)} \leftarrow agg(sample\ right\ item\ neighbours)$
    $h_r^k \leftarrow \sigma(W_r^k.COMBINE(h_r^{(k-1)}, h_{L(r)}^{(k-1)}, h_{R(r)}^{(k-1)}) + b)$
  $score_{(u,v)} \leftarrow \sigma(f(h_u^K, h_v^K))$

Chapter 5

Experiments

Here we will evaluate SESTKGCN model on three datasets on real life scenarios, namely movie, music, and book recommendations and compare it against other famous GCN models.

5.1 Datasets

We took the following datasets in consideration for movie, music and books respectively.

- **MovieLens-20M** consists of approximately 20 million explicit ratings (ranging from 1 to 5) on the MovieLens website.
- **Last.FM** contains musician listening information 2000 users from Last.fm online music system.
- **Book-Crossing** contains 1 million ratings ranging from 0 to 10 of books in the Book-Crossing community.

We transform the explicit feedbacks in the three into implicit feedback where each entry is marked with 1 indicating that the user has rated the item positively, and sample an unwatched set of items marked as 0 for each user and the size of this set is equal to size of positive item set for this user. The threshold of positive rating is 4 for MovieLens-20M, 0+ for BookCrossing and Last.FM due to their sparsity.

For movie dataset we took all the tags to a movie with relevance ¿ 0.9 and then connect items with same tag and label the relationship between them with the current tag. We also got the time information when an item was rated in this dataset. But we did not get a social graph.

For music dataset we took the count of number of users who marked a atrist with a particular tag. Then we took tags to an artist as relevant if it's user count is greater than 5. Here we also have user social network.

For book dataset we took just two type of relationship between books and that is "author of book" and "title given to that book". So those who have same author or title is connected by an edge. Here we got the user social network graph by connecting the users whose age were in a difference of 5 and also if the user belongs to same city. If user is connected by age the weight of the edge was kept as 3 and if the user is connected by same city then it is 1.

We also took out one item for each user from its positive set to keep it in test set to get the top K hit rate. Then we divided the rest of user item pair into train(0.8) and validation(0.2) set.

We initialised the embeddings of each of the user, item and relationship using normal initialization with mean 0 and standard deviation to 0.8 and kept the embeddings to trainable mode(requires gradient = True).

We also conducted experiments with a different initialization of embeddings. Here we took the whole user-item graph G and made each relationship as a node and then applied nodetovec approach using biased random walk with walk length=10, p = 2, q = 0.5, num walks=40. Using this initialization surely improved the results. Here also we kept the embeddings to trainable.

The basic statistics of the three datasets are presented in Table 5.1. We have cropped some users for movie and book datasets for fast training.

TABLE 5.1: Basic statistics and hyper-parameter settings for the three datasets (K: neighbor sampling size, d: dimension of embeddings, L: number of layers, batch: batch size, λ: L2 regularizer weight, η: learning rate).

stat	movie	music	book
users	20175	1892	3000
items	102569	1511	60390
relationships	32	258	2
user-item interactions	1967210	113836	73718
K	32	32	32
d	32	16	32
L	1	1	1
Batch	1000	1000	256
λ	2e-4	2e-3	2e-3
η	2e-3	2e-2	5e-3

5.2 Baselines

We compared the results of SESTKGCN against KGCN approach.

- **KGCN** uses knowledge graph information to aggregate embeddings of items while the embedding of user is kept the same as input embedding. Gets the output probability by doing a dot of user embedding to item final aggregated embedding.

5.3 Experimental Setup

In SESTKGCN we have used inner product while calculating the importance of item by multipling the embedding of current item to relationship embedding. While using the concept of spatial GCN for time and position we used tanh as non linearity. Also while aggregating at each layer after passing the input aggregations through Weight matrix we used tanh as the nonlinearity again. After doing the final dot product we apply sigmoid on each output to get the probability between 0 and 1. I have kept one item for each user as test set and divided the rest interactions as 80% for train and 20% for validation. The code is in pytorch.

The hyper parameters for KGCN was kept same as mentioned in KGCN paper with batch size 1000.

We evaluate our model on two scenarios. (i)One is we apply our model to predict the interaction for user item pairs in test set and get the F1 score and the area under curve(auc). (ii)Another is, we try to find the hit rate for each user, which is the number of users for which the item in test set is also found in top-K items predicted by our model.

5.4 Results

The result for F1 score and auc for SESTKGCN and KGCN is presented in Table 5.2 and the top-K hit rate is shown in Table 5.3.

We see that the improvement of SESTKGCN is quite significant on music dataset. We got much better AUC and F1 scores than KGCN as per our results of KGCN on music dataset and also better than what is mentioned in KGCN paper. This is a great outcome since Last.FM are much sparser than MovieLens20M.

Table 5.2: F1 score and AUC in interaction prediction.

	movie		music		book	
model	AUC	F1	AUC	F1	AUC	F1
KGCN-sum	0.9663	0.9653	0.6256	0.6135	0.6380	0.5880
KGCN-concat	0.9521	0.9438	0.5992	0.5541	0.6123	0.5811
SESTKGCN-concat	0.9790	0.9421	0.9012	0.8180	0.5206	0.5320
SESTKGCN-sum	0.9598	0.9283	0.8836	0.7894	0.5000	0.6644
SESTKGCN-nodetovec-concat	-	-	0.8997	0.8212	0.5180	0.5520

We also see that even on movie dataset SESTKGCN does better or atleast same as KGCN. The reason for nearly same outcome could be that movie dataset does not have social graph and so its taking side information, only from Knowledge Graph. But still it shows some improvement.

On book dataset KGCN-sum works well.

5.4.1 Impact of dimension of embedding

Varying the dimensions of the embeddings, we saw that higher dimensions were not giving such good results on music dataset. As the amount of information is too less, we observed that with increasing dimension train loss decreases a lot but test loss starts increasing. This may be because the model was memorising the information rather than generalizing. We took several steps to avoid over fitting, such as, increased regularization, incorporated batch normalization, decreased the dimension, decreased the number of layers as we saw increasing layers was not giving such good results on music dataset.

5.4.2 Impact of number of layers

We say that increasing the layer on music and book dataset was not showing much improvement rather was giving bad performance. This was maybe due to over parametrization.

TABLE 5.3: hit rate for top-K prediction.

Dataset	top-10	top-50	top-100
movie	0.0001	0.0026	0.0118
music	0	0.0038	0.0102
book	0	0.0006	0.0016

Chapter 6

Conclusions and Future work

6.1 Conclusion

SESTKGCN is able to extract information from item Knowledge graph, social network, and user position and time for recommendation very well. It is able to learn graph structural information and perdict users potential interests. Also it can be called better than KGCN as KGCN gets different item aggregation embedding for the same item if the user is different, i.e., the item embeddings depends on user and so if there are so many users. We would have to calculate item embeddings so many times for the same item. Whereas in SESTKGCN you can get all the users and item embeddings in one go as item embeddings is not dependent on the user it is calculating for. After getting all the embeddings you can simply do a dot of user and item embedding for the pair you want to predict the rating and so you do not have to aggregate item embedding for each user.

6.2 Future Work

We propose below variations that can be tried with our model.

- Try to experiment more using node2vec and other unsupervised approaches to initialise user, item and relationships embeddings.
- Try adding user bias and item bias to the output scores for user item pair.
- Try pairwise training in a way similar to Bayesian Personalized Ranking, where we take one positive sample and one negative sample together at a time while training. Also use the loss function same as Group Preference Based Bayesian Personalized Ranking(GBPR) i.e., the log-likelihood of GBPR.

Bibliography

[1] Wang, Hongwei and Zhao, Miao and Xie, Xing and Li, Wenjie and Guo, Minyi, "Knowledge Graph Convolutional Networks for Recommender Systems" in WWW '19. Available: https://arxiv.org/pdf/1904.12575.pdf

[2] Shiwen Wu and Fei Sun and Wentao Zhang and Bin Cui, "Graph Neural Networks in Recommender Systems: A Survey", 2021. Available: https://arxiv.org/pdf/2011.02260.pdf

[3] Tomasz Danel and Przemysław Spurek and Jacek Tabor and Marek Śmieja and Łukasz Struski and Agnieszka Słowik and Łukasz Maziarka, "Spatial Graph Convolutional Networks", 2020. Available - https://arxiv.org/pdf/1909.05310.pdf

[4] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner and Lars Schmidt-Thieme, "BPR: Bayesian Personalized Ranking from Implicit Feedback" in UAI 2009. Available: https://arxiv.org/ftp/arxiv/papers/1205/1205.2618.pdf

[5] Wang, Xiang and He, Xiangnan and Cao, Yixin and Liu, Meng and Chua, Tat-Seng, "KGAT" in ACM SIGKDD 2019. Available: https://arxiv.org/pdf/1905.07854.pdf

[6] Xiao Sha and Zhu Sun and Jie Zhang, "Attentive Knowledge Graph Embedding for Personalized Recommendation", 2020. Available: https://arxiv.org/pdf/1910.08288.pdf

[7] Federico Bianchi, Gaetano Rossiello, Luca Costabello, Matteo Palmonari, Pasquale Minervini, "Knowledge Graph Embeddings and Explainable AI", 2020. Available: https://arxiv.org/pdf/2004.14843.pdf

[8] Shenglin Zhao and Irwin King and Michael R. Lyu, "A Survey of Point-of-interest Recommendation in Location-based Social Networks", 2016. Available: https://arxiv.org/pdf/1607.00647.pdf

[9] Wang, Xiang and He, Xiangnan and Cao, Yixin and Liu, Meng and Chua, Tat-Seng, "IntentGC", 2019. Available: https://arxiv.org/pdf/1907.12377.pdf

[10] Jiani Zhang and Xingjian Shi and Shenglin Zhao and Irwin King, "STAR-GCN: Stacked and Reconstructed Graph Convolutional Networks for Recommender Systems", 2019. Available: https://arxiv.org/pdf/1905.13129.pdf

[11] Yu, Bing and Yin, Haoteng and Zhu, Zhanxing, "Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting", 2018. Available: https://arxiv.org/pdf/1709.04875.pdf

[12] William L. Hamilton and Rex Ying and Jure Leskovec, "Inductive Representation Learning on Large Graphs", 2018. Available: https://arxiv.org/pdf/1706.02216.pdf

[13] Liu, Yanchi and Liu, Chuanren and Liu, Bin and Qu, Meng and Xiong, Hui, "Unified Point-of-Interest Recommendation with Temporal Interval Assessment", KDD 2016. Available: https://www.kdd.org/kdd2016/papers/files/rfp0482-liuA.pdf

[14] Jure Leskovec, "Graph Representation Learning". Available: http://snap.stanford.edu/class/cs224w-2018/handouts/09-node2vec.pdf

[15] Yehuda Koren, Robert Bell and Chris Volinsky, "Matrix Factorization Techniques for Recommender Systems". Available: https://datajobs.com/data-science-repo/Recommender-Systems-

[16] Pan, Weike and Chen, Li, "GBPR: group preference based Bayesian personalized ranking for one-class collaborative filtering", IJCAI 2013. Available: tinyurl.com/42rtuk3v

[17] 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011) Datasets. Available: https://grouplens.org/datasets/hetrec-2011/

[18] DGL - Deep Graph Library, "Inductive Representation Learning on Large Graphs (GraphSAGE)" - Pytorch implementation. Available: https://github.com/dmlc/dgl/tree/master/examples/pytorch/graphsage/experimental

[19] Cai-Nicolas Ziegler, "Book-Crossing Dataset". Available: http://www2.informatik.uni-freiburg.de/ cziegler/BX/

[20] Yelp Dataset, Available: https://www.kaggle.com/yelp-dataset/yelp-dataset?select=yelp_academic_dataset_business.json

[21] Fabrizio Frasca and Emanuele Rossi and Davide Eynard and Ben Chamberlain and Michael Bronstein and Federico Monti, "SIGN: Scalable Inception Graph Neural Networks", 2020. Available: https://arxiv.org/pdf/2004.11198.pdf

[22] Priyanka Gupta, Diksha Garg, Pankaj Malhotra, Lovekesh Vig, and Gautam M. Shroff, "NISER: Normalized Item and Session Representations with Graph Neural Networks", 2019. Available: https://arxiv.org/pdf/1909.04276.pdf

[23] Zhixiang He, Chi-Yin Chow, and Jia-Dong Zhang, "GAME: Learning Graphical and Attentive Multi-View Embeddings for Occasional Group Recommendation", 2020. Available: https://dl.acm.org/doi/10.1145/3397271.3401064