B-Tech. Term Project

Socially Enhanced Spatial Temporal Knowledge Graph Convolutional Network

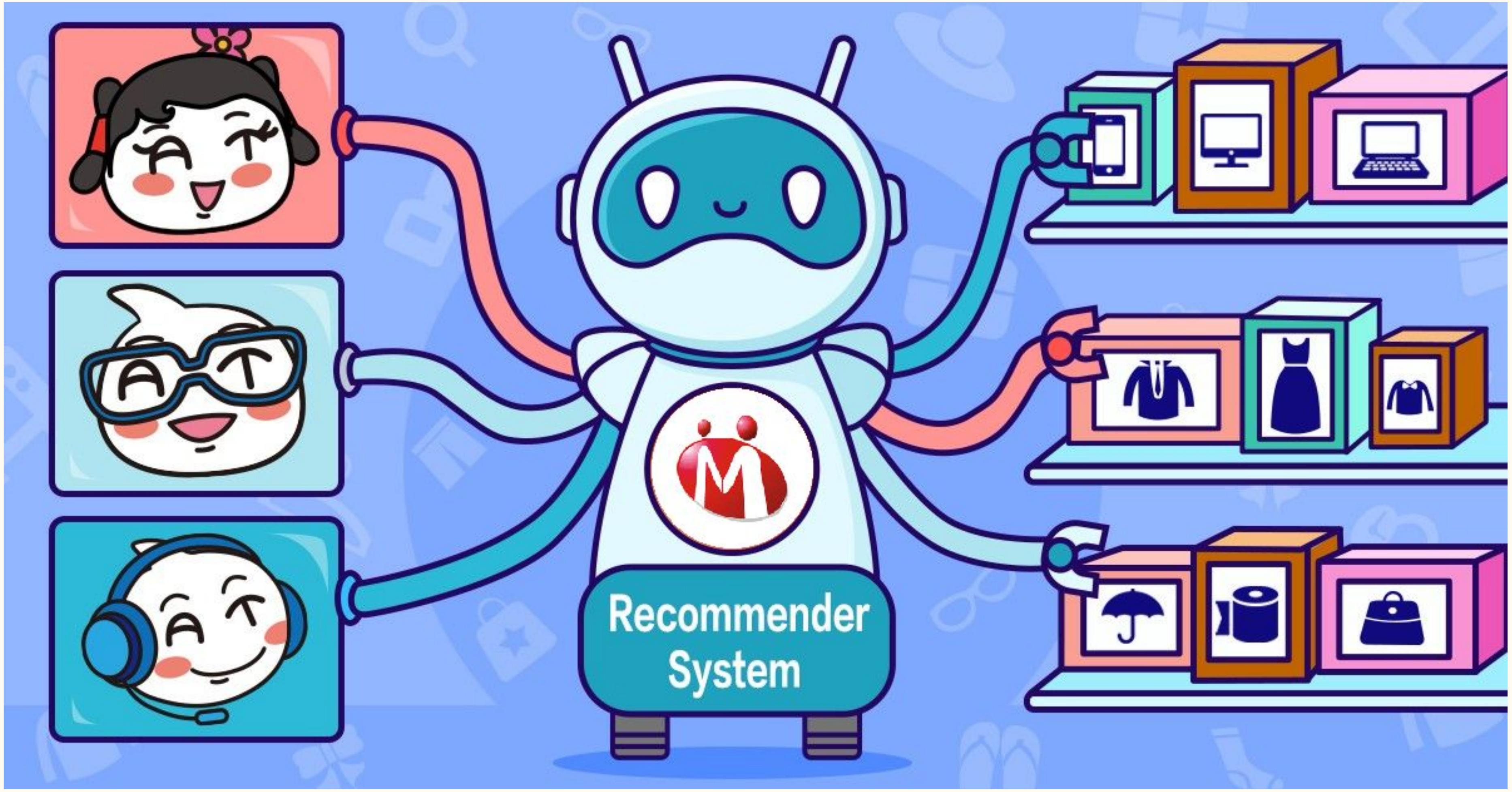
Under the Guidance of:

Prof. Pabitra Mitra

Submitted By:

Jyoti Agrawal

(17CS10016)


Recommender System

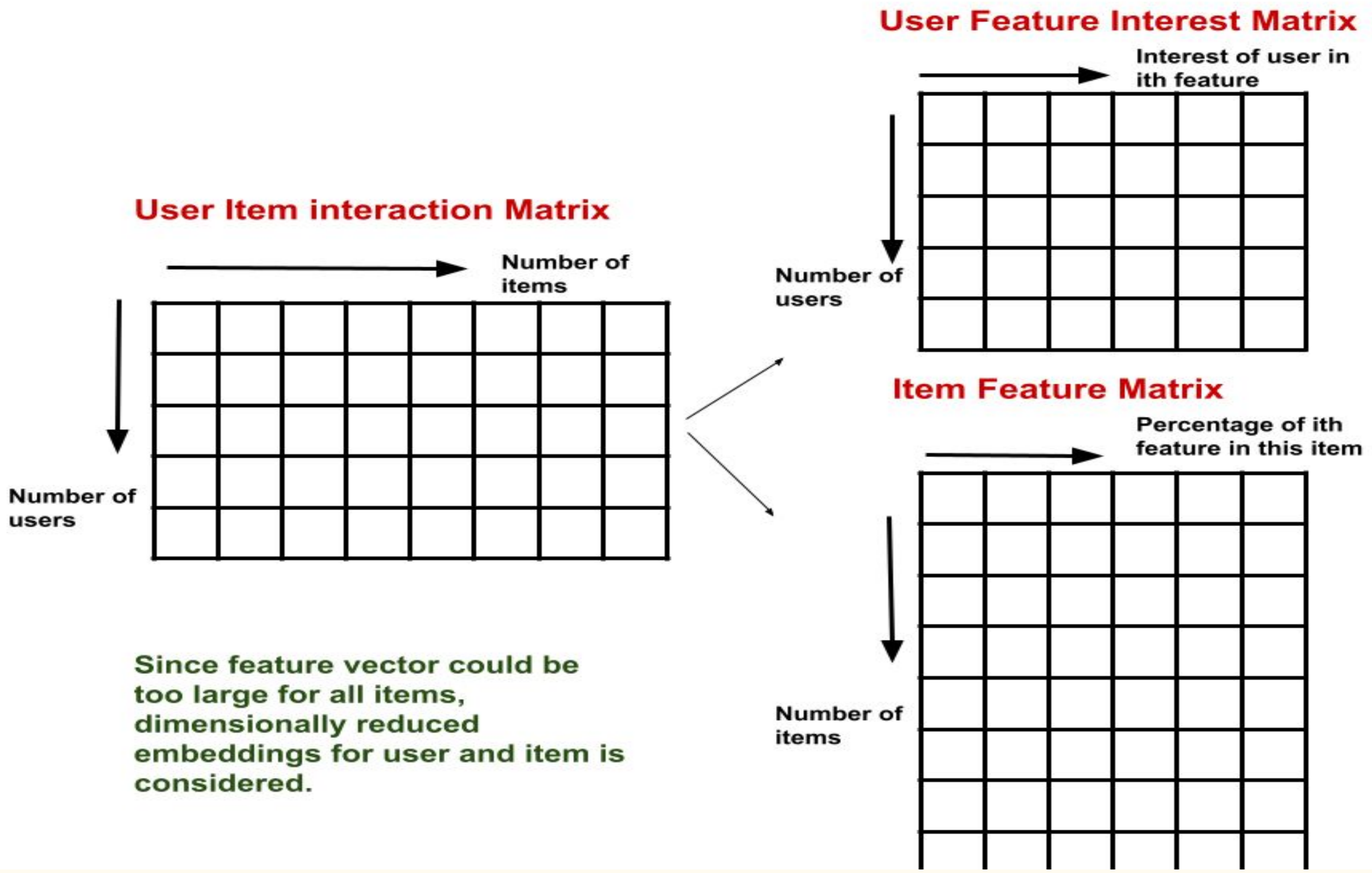
Motivation

- ➔ Recommender system helps user discover items that they like.
- ➔ Today with the internet reaching nearly everyone, recommendation system has become a very important and integral part of online trade.
- ➔ Examples are, e-commerce websites, online movie sites, news, real estate websites, tourism websites, etc.
- ➔ 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 information such as social relationship and knowledge graph is the main challenge in this field.
- ➔ We need to leverage graph learning approaches in this field as most information available has graph structure.

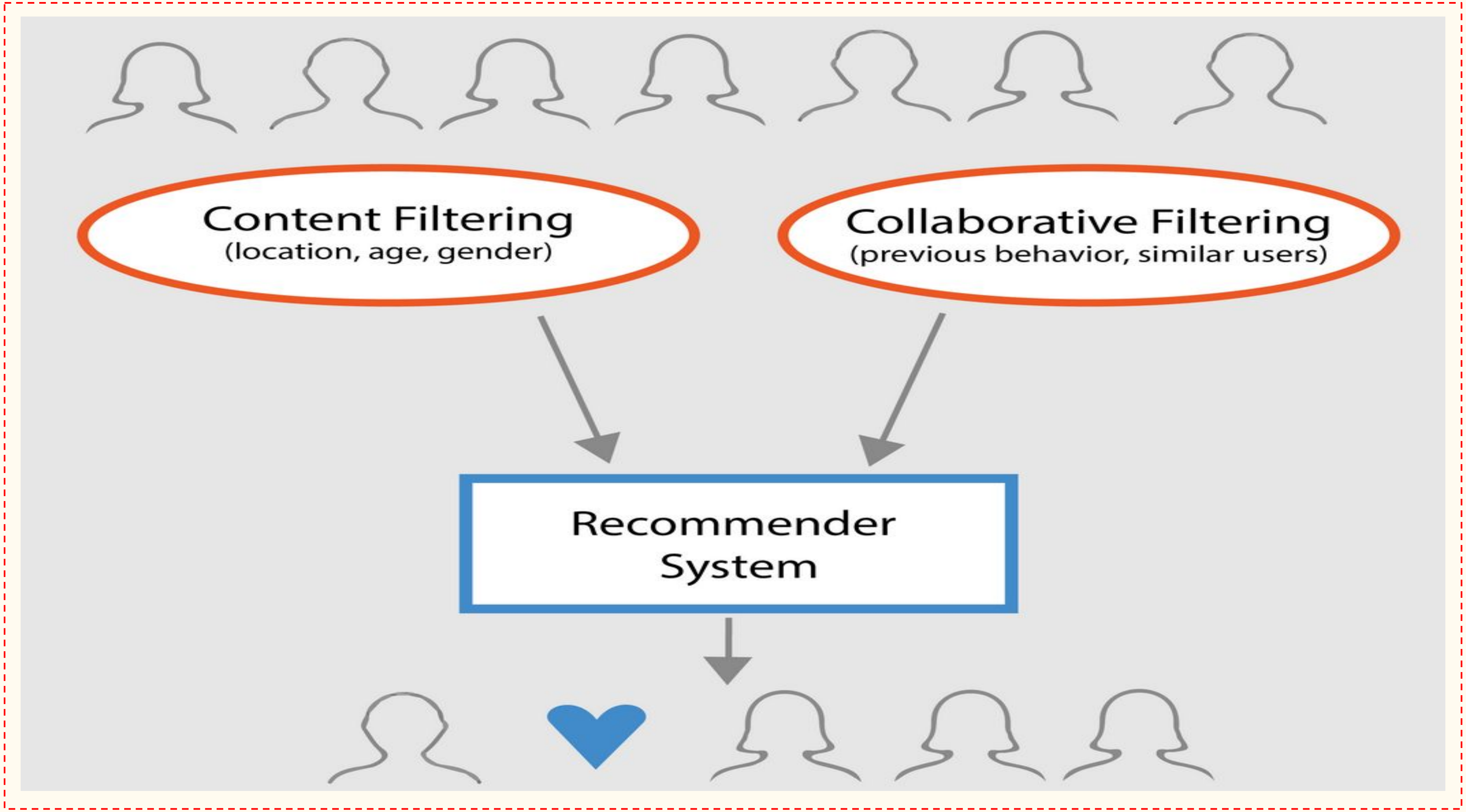
Recommendation Systems: How Do They Work

Introduction

- ➔ This algorithm uses GCN based approach to aggregate 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 G. You have input feature vectors of each entity node(user / item) and input vectors corresponding to relationships in item Knowledge.
- ➔ Also assumes that we know users position vector and the time when a user interacted with a particular item.
- ➔ This algorithm also uses concept of Spatial Graph Convolution (Spatial GCN) to take care of the Position of user and time of user item interaction.
- ➔ The baseline concept of this approach is same as matrix factorization.
- ➔ The task is to learn embeddings based of the available user item interactions and to predict the probability of a user being interested in an unseen item.


User Item interaction Matrix
Number of items
Number of users
User Feature Interest Matrix
Interest of user in ith feature
Number of users
Item Feature Matrix
Percentage of ith feature in this item
Number of items

Since feature vector could be too large for all items, dimensionally reduced embeddings for user and item is considered.


Content Filtering
(location, age, gender)
Collaborative Filtering
(previous behavior, similar users)
Recommender System

Related Work

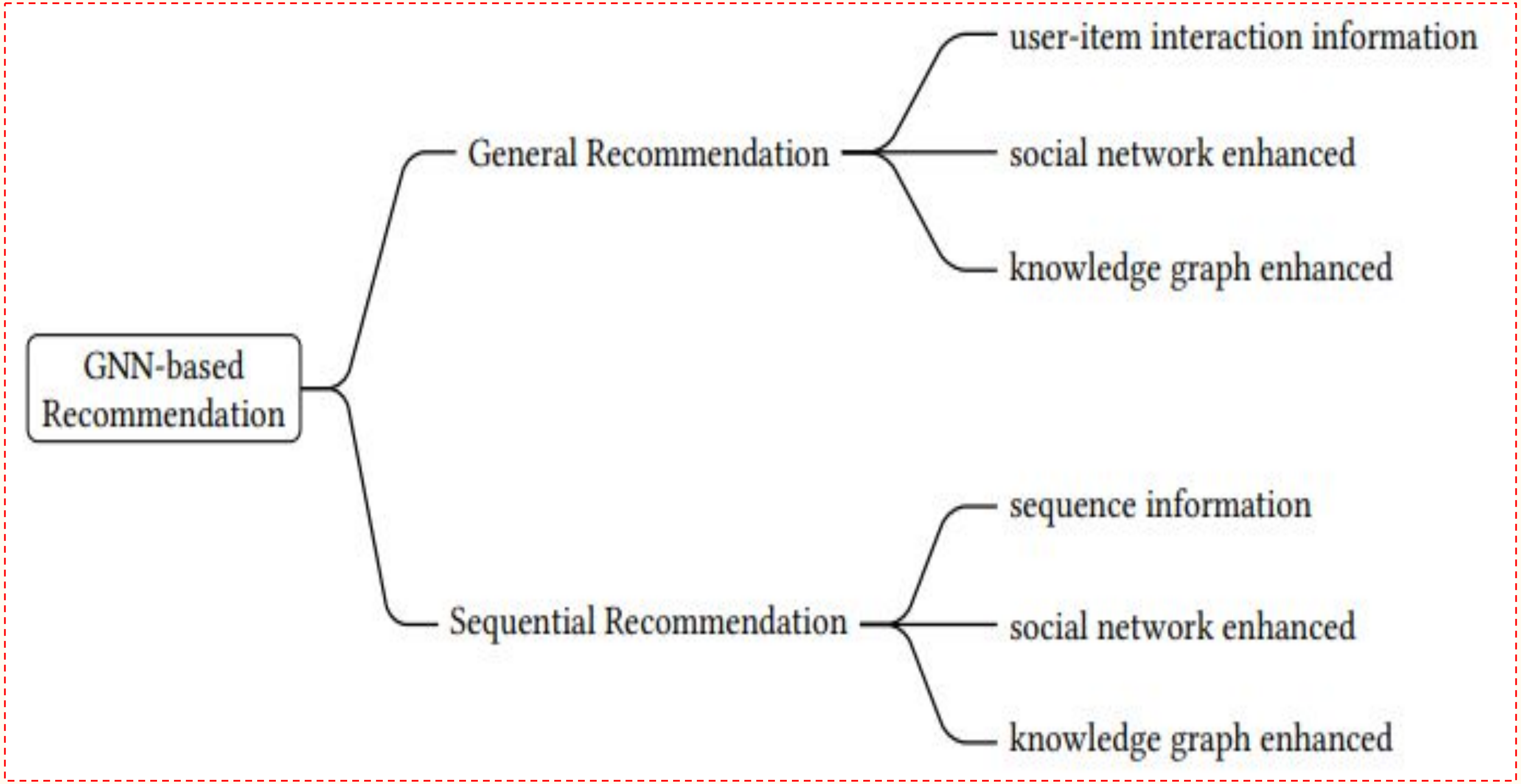
- Some early methods used item based similarity of the historical items taken by the user to recommend new items.
- But today there is alternative approach that tries to encode both items and users as embeddings in a common space (Representation based model) from matrix factorization to deep learning(GNN) models.

General Recommendation

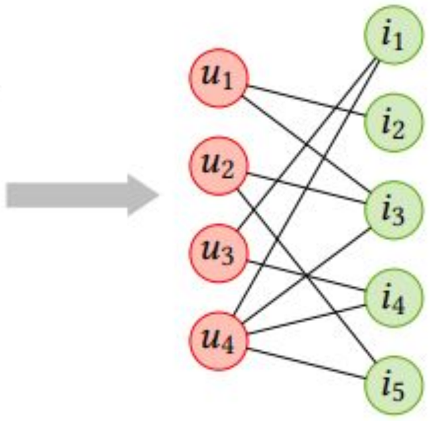
Given an item set I, the user set U, and the observed interactions between users and items R : U x I, the task is to estimate for any user u ∈ U his/her preference for any item i ∈ I by the representation hu and hi learnt for user and items representation, i.e. $y_{u,i} = f(h_u^*, h_i^*)$

Sequential Recommendation

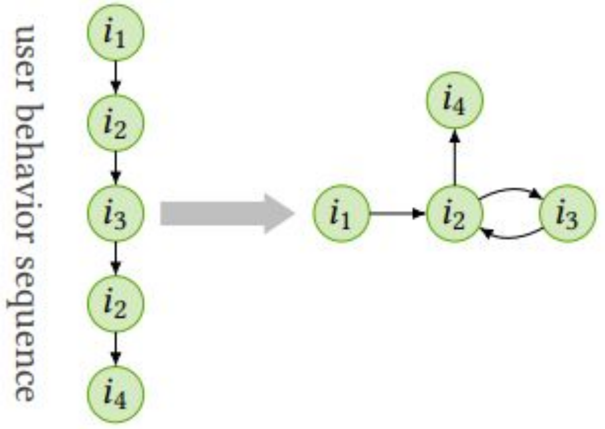
Having sequence of items interacted by a user ordered in terms of time, the task is to predict the next item a user might be interested in.


GNN-based Recommendation
General Recommendation
user-item interaction information
social network enhanced
knowledge graph enhanced
Sequential Recommendation
sequence information
social network enhanced
knowledge graph enhanced

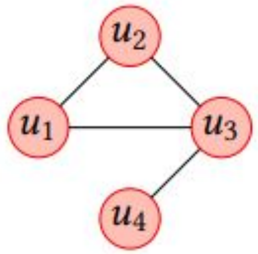
	i_1	i_2	i_3	i_4	i_5
u_1	0	1	1	0	0
u_2	0	0	1	0	1
u_3	1	0	0	1	0
u_4	1	0	1	1	1



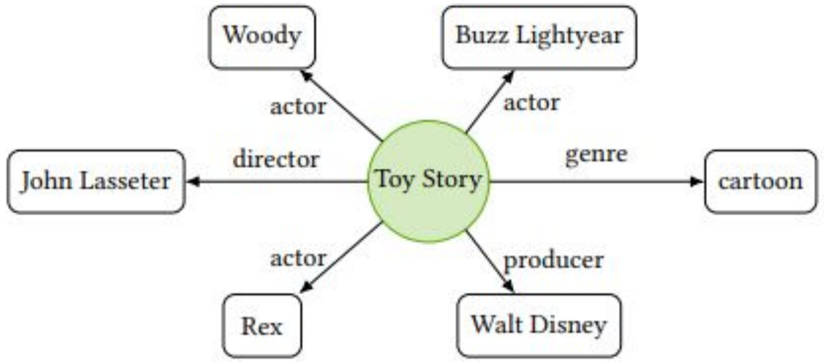
(a) User-item bipartite graph.



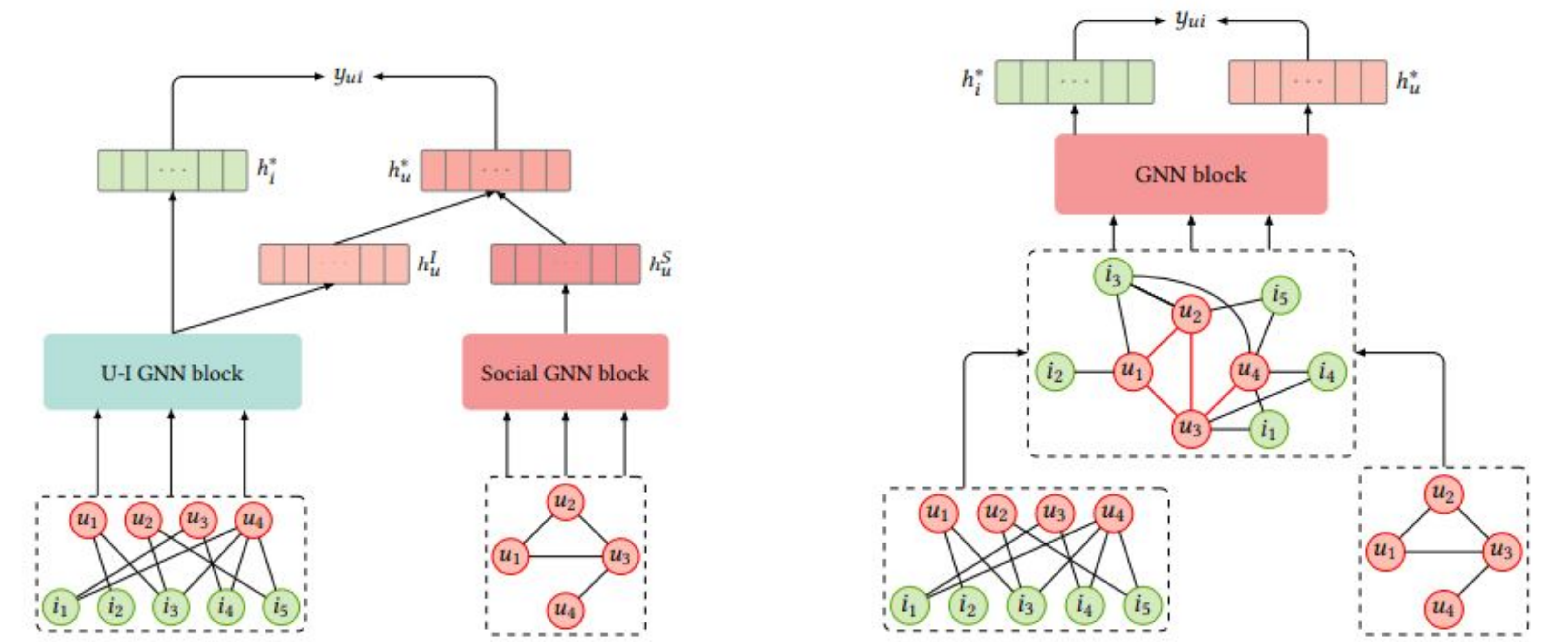
(b) Sequence graph.



(c) Social relationship between users.



(d) Knowledge graph



(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.

Example of Social Network Enhanced Recommendation Model

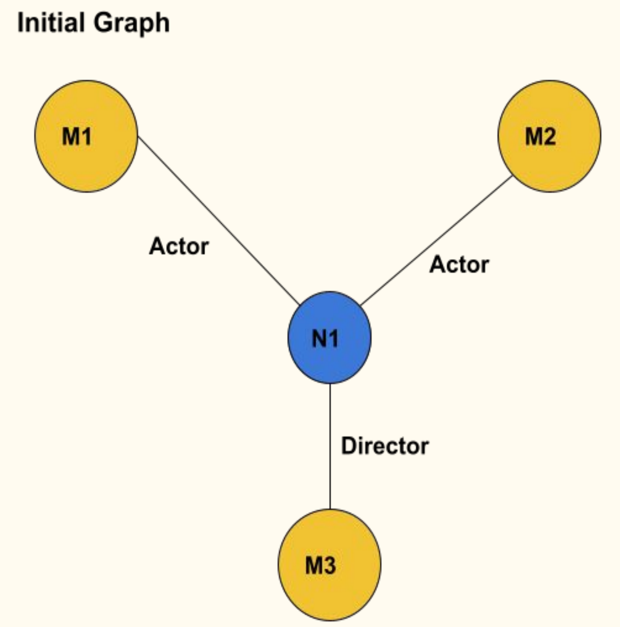
Contribution

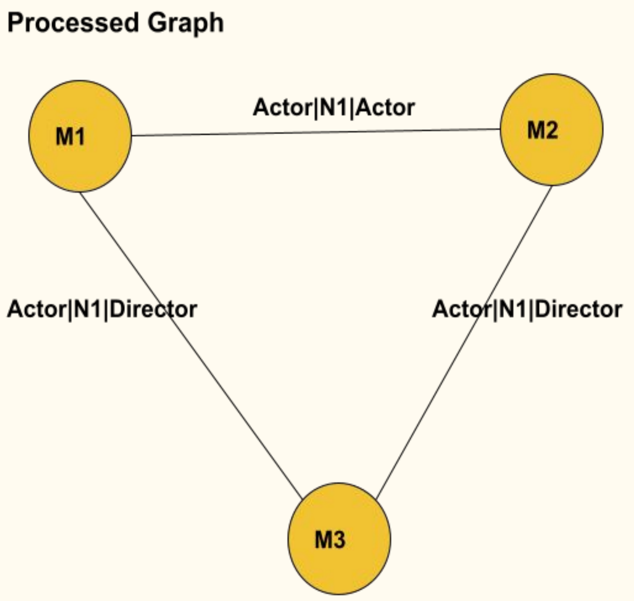
- Proposed an end-to-end model for recommendation incorporating the item Knowledge Graph, User Social Network Graph and the User Item interaction bipartite graph.
- Showed the results of our model on three real life recommendation scenarios, i.e., movie, music and books.
- Released the code of SESTKGCN and the datasets at `https://github.com/jyoti246/SESTKGCN` for validation and future work.

Algorithm

Graph Preprocessing

- KG-Preprocessing





Graph Preprocessing

➔ Combined User-Item Graph can be thought of a heterogeneous graph which is realization of combination of the below three graphs with edges containing different information in different Graphs

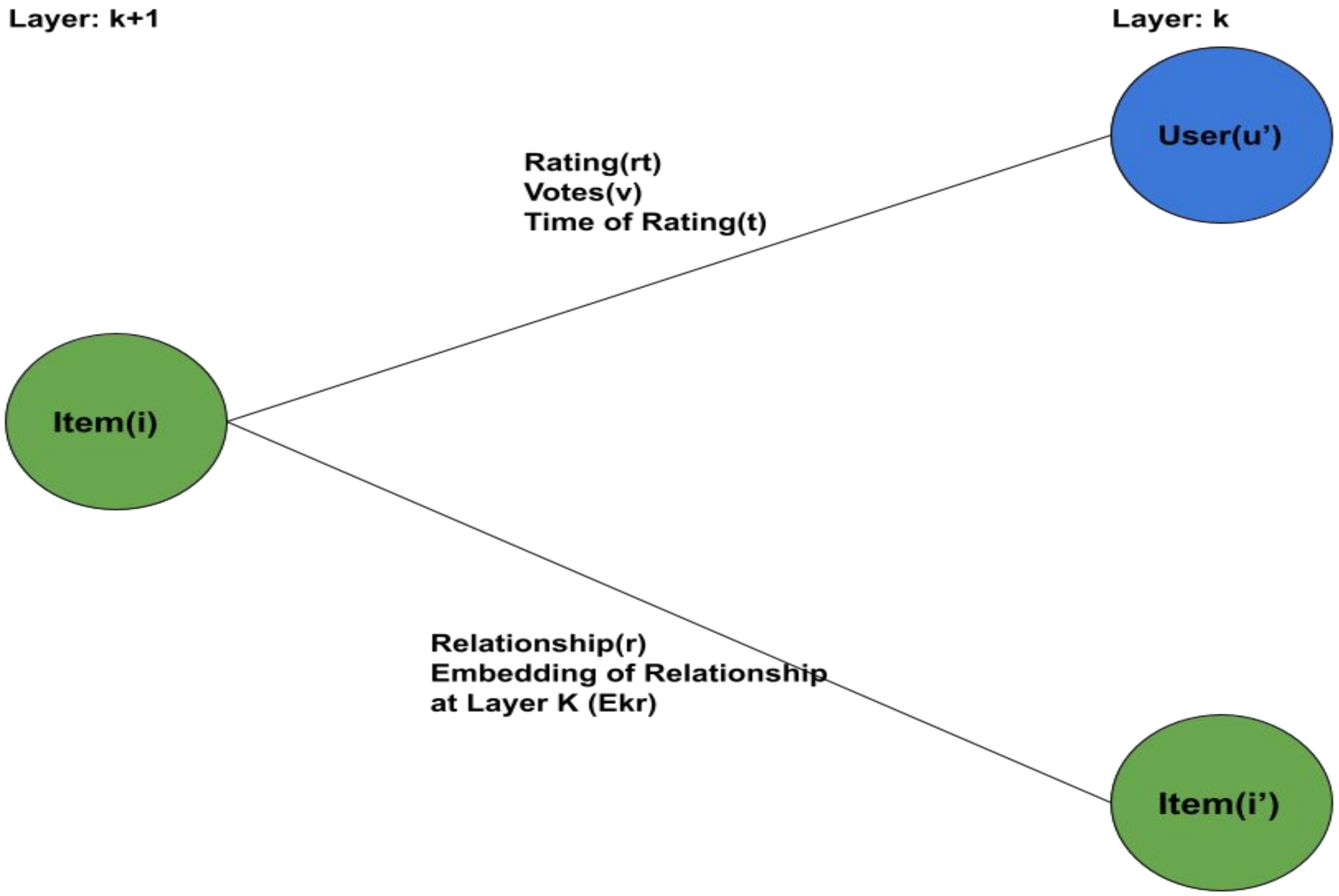
1. Item Knowledge Graph with edges as item-item relationship
2. User Social Network having strength of user user interaction on each edge
3. User-Item bipartite graph with rating, number of votes to a rating and time of rating information available at each edge.

Problem Formulation

- We have a set of M users U = $u_1, u_2, \ldots, u_m$ and a set of N items V = $v_1, v_2, \ldots, v_n$.
- We have the user item interaction matrix Y which is based of users feedback to a particular item, i.e, 1 if the the feedback is positive and 0 if negative.
- Additionally we have the User-Item Graph as a combination of the 3 graphs as explained above.
- We also have user position information and time of interaction a user and item.
- Our goal is to learn a prediction function $y_{u,v} = F(u, v | \theta, Y, G)$, where $y_{u,v}$ denotes the probability that user u will engage with item v, and θ denotes the model parameters of function F.

Methodology

- 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(0)u, h(0)i and h(0)r respectively.
- We also have position of each user (pu), strength of friendship between u and u’ (s(u,u’)), rating given by user u to item i (rt(ui)) and the number of votes given to this rating (v(ui)) and time (t(ui)) 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.


Layer: k+1
Layer: k
User(u')
Rating(rt)
Votes(v)
Time of Rating(t)
Item(i)
Relationship(r)
Embedding of Relationship
at Layer K (Ekr)
Item(i')

Updation of Item node Embedding

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

OR

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

where

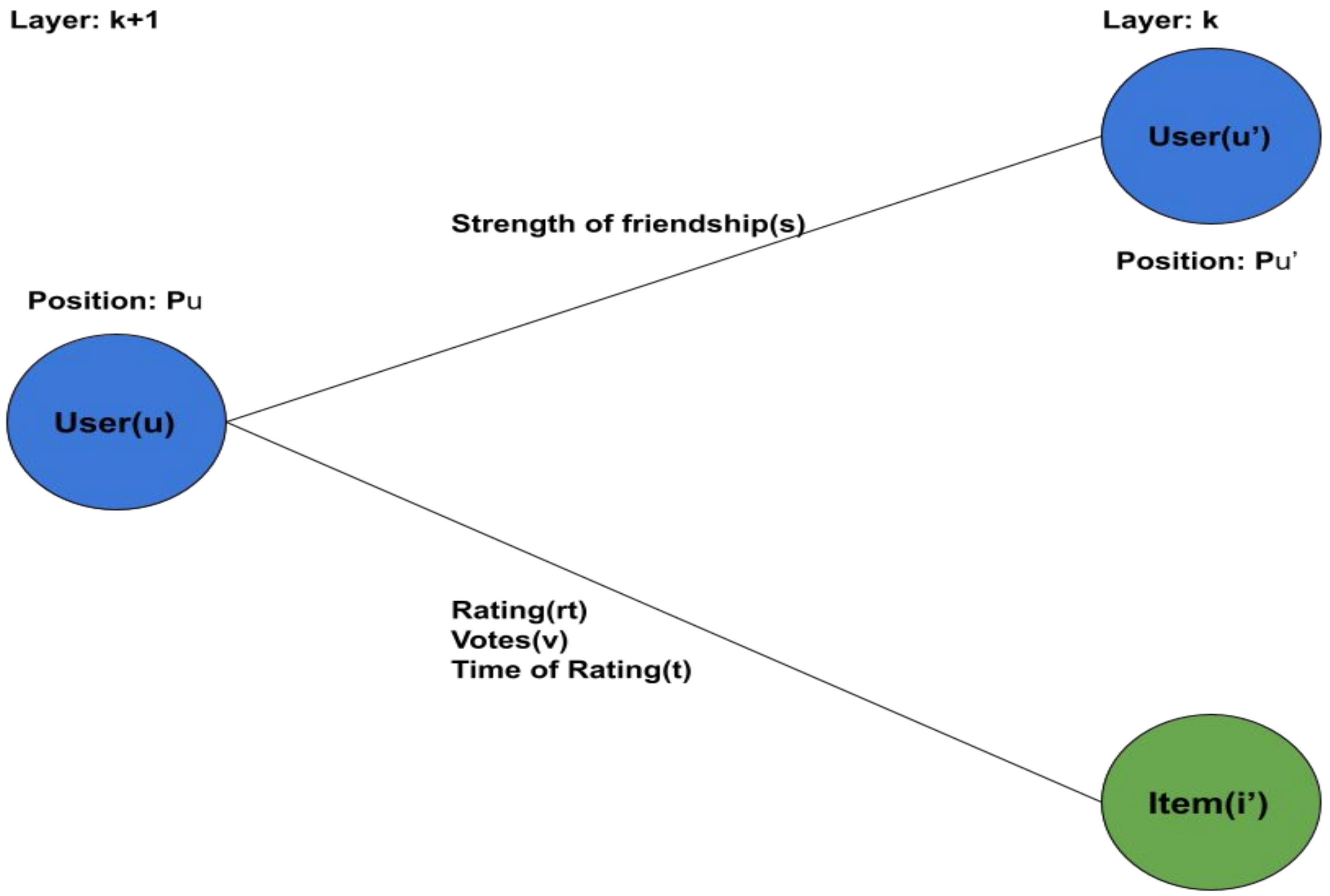
$$h_{N^U(i)}^{(k)} = \sum_{u' \in N^U(i)} rt_i^{u'} * v_i^{u'} * (EMB(t_i^{u'}) \odot h_{u'}^{(k)})$$

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

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

where $\pi_{r_i^{i'}}^{i} = g(i, r_i^{i'})$

and $\pi_{r_i^{i'},NORM}^{i} = \dfrac{exp(\pi_{r_i^{i'}}^{i})}{\sum_{i" \in N^I(i)} exp(\pi_{r_i^{i"}}^{i})}$


Layer: k+1
Layer: k
User(u')
Strength of friendship(s)
Position: Pu'
Position: Pu
User(u)
Rating(rt)
Votes(v)
Time of Rating(t)
Item(i')

Updation of User node Embedding

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

OR

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

where

$$h_{N^U(u)}^{(k)} = \sum_{u' \in N^U(u)} s_u^{u'} * (EMB'(pos_u^{u'}) \odot h_{u'}^{(k)})$$ where $$pos_u^{u'} = |p_{u'} - p_u|$$

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

$$h_{N^I(u)}^{(k)} = \sum_{i' \in N^I(u)} rt_{i'}^u * v_{i'}^u * (EMB(t_{i'}^u) \odot h_{i'}^{(k)})$$ where $$EMB(t) = \sigma(U^k.t + b)$$

Updation of item-item Relationship Embedding

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

OR

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

where

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

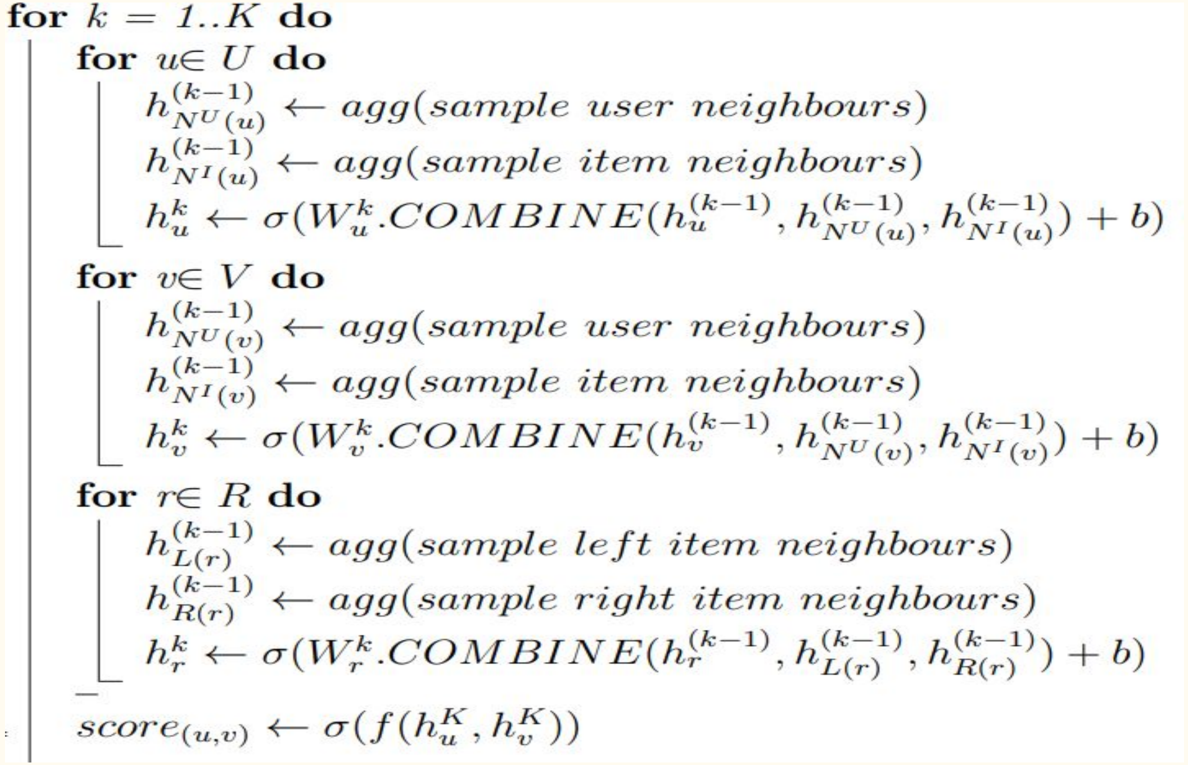
$$h_{R(r)}^{(k)} = AGGREGATE(h_i^{(k)}, FOR[i \in R(r)])$$

Learning 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 as:

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

where $f : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$



```
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
        $h_{L(r)}^{(k-1)} \leftarrow agg(sample\ left\ item\ neighbours)$
        $h_{R(r)}^{(k-1)} \leftarrow agg(sample\ right\ item\ neighbours)$
        $h_r^k \leftarrow \sigma(W_r^k.COMBINE(h_r^{(k-1)}, h_{L(r)}^{(k-1)}, h_{R(r)}^{(k-1)}) + b)$
    $score_{(u,v)} \leftarrow \sigma(f(h_u^K, h_v^K))$
```

Experiments

Datasets

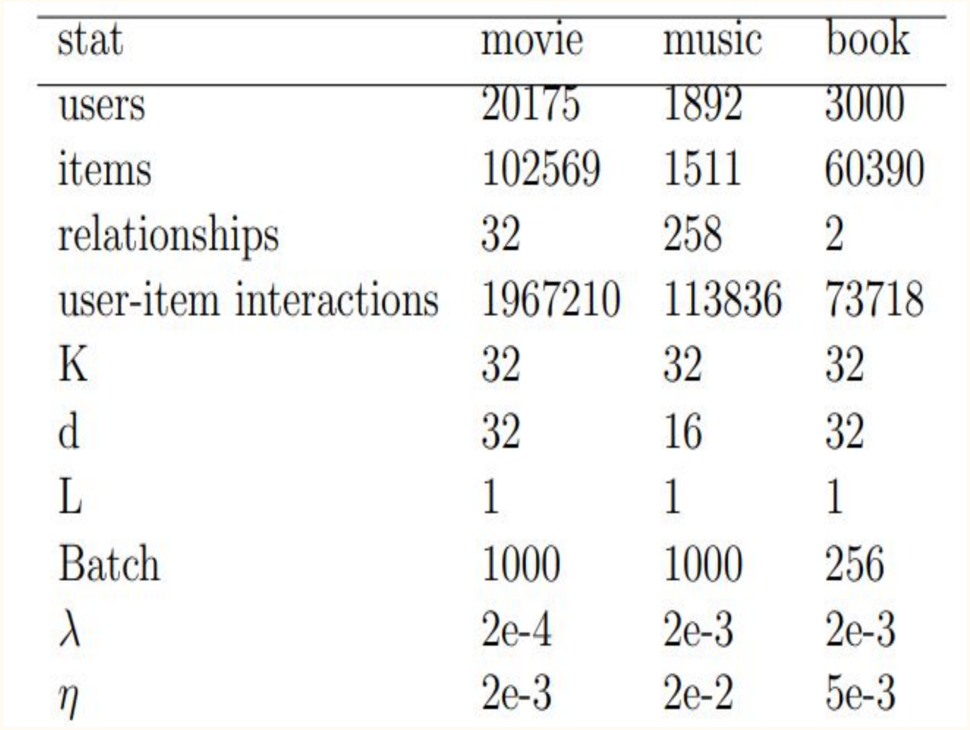
- **MovieLens-20M** consists of approximately 20 million explicit ratings (ranging from 1 to 5) on the MovieLens website.
- **Last.FM** contains musician listening information 2000 users from Last.fm online music system.
- **Book-Crossing** contains 1 million ratings ranging from 0 to 10 of books in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

Setup

- ➔ Transformed explicit feedback into implicit where each entry is marked 1 indicating positive interaction, and sample an unwatched set of items as 0 for each user and the size of this set is equal to size of positive item set for this user.
- ➔ The threshold for positive interaction is kept at 4 for movie dataset and 0+ for music and book dataset due to their sparsity.
- ➔ For movie dataset, took all the tags to a movie with relevance > 0.9 and connected items with same tag with tag name as relationship. No social network graph.
- ➔ For music dataset, took the tags to an artist as relevant if more than 5 users put it. Social graph also available.
- ➔ For book dataset, took author and the title as the only relationship. Created social graph by connecting users with age difference of 5 or those living in same city.

Setup

- ➔ Took 1 item from each users' positive set and kept it to test set to get the top-K hit rate. Divided rest of the data into train(80%) and validation set(20%).
- ➔ Initialised embeddings to normal distribution keeping embeddings to trainable mode(requires gradient = True).
- ➔ Also conducted experiments using nodetovec approach capturing macroscopic view for embedding initialization.



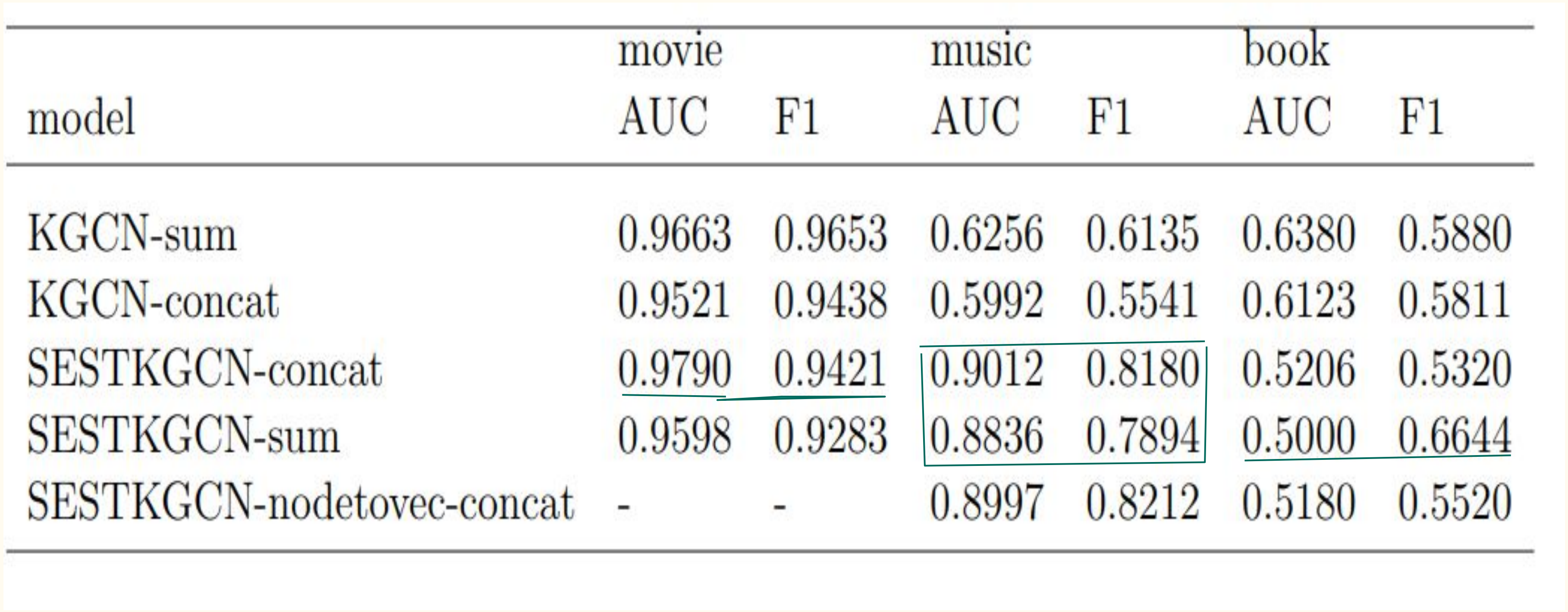
stat	movie	music	book
users	20175	1892	3000
items	102569	1511	60390
relationships	32	258	2
user-item interactions	1967210	113836	73718
K	32	32	32
d	32	16	32
L	1	1	1
Batch	1000	1000	256
λ	2e-4	2e-3	2e-3
η	2e-3	2e-2	5e-3

Baseline

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.

Results



model	movie AUC	movie F1	music AUC	music F1	book AUC	book 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

Results

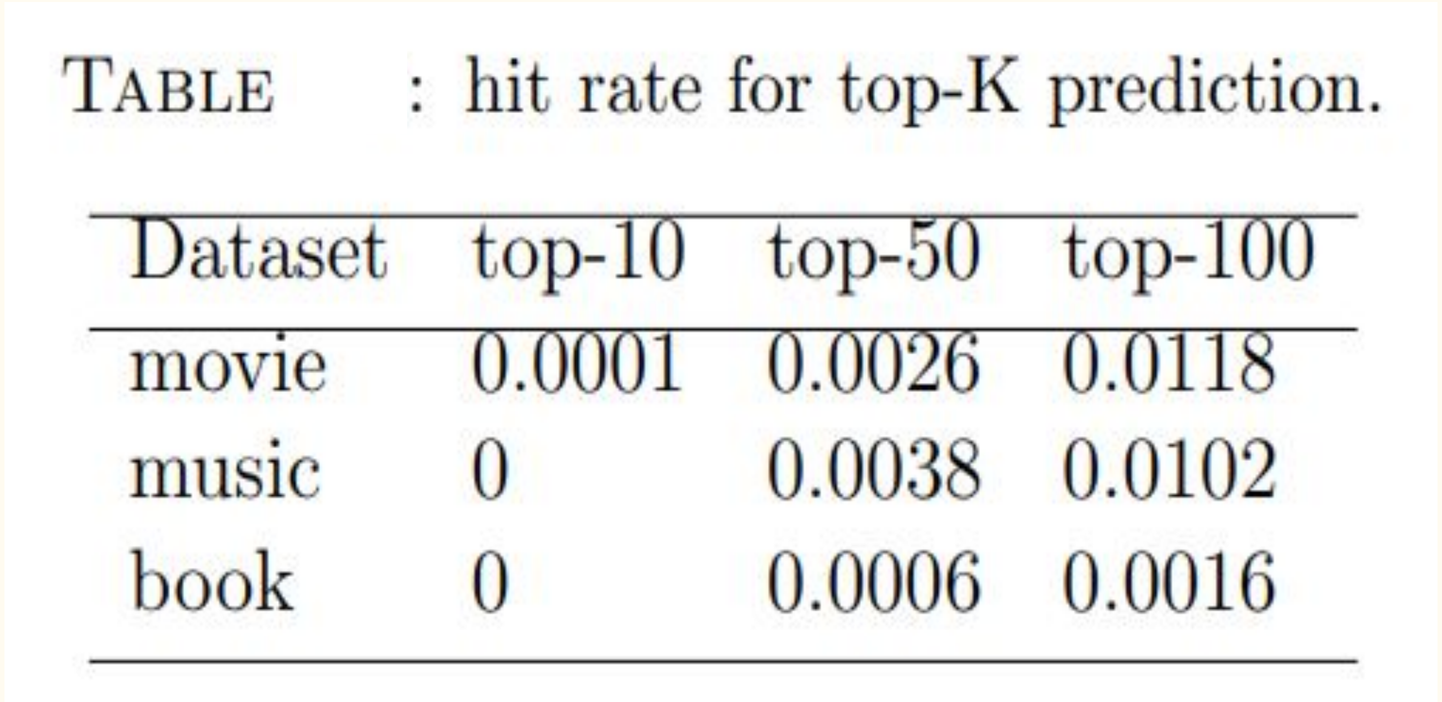


TABLE : hit rate for top-K prediction.

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

Difficulties and Observations

- ➔ Over fitting because of Over Parameterization due to large number of item-item relationships, i.e, test loss was increasing with the decrease of train loss.
- ➔ Reduced the number of relationships to most important ones, used batch normalization and varied the batch size.
- ➔ Also decreased embedding dimensions and number of layers.
- ➔ Also tried changing the sampling from dynamic to static.

Conclusion

- SESTKGCN is able to extract information from item Knowledge graph, social network, and user position and time for recommendation.
- It is better than KGCN as KGCN gets different item aggregation embedding for the same item if the user is different.
- It shows **better results**.

Future Work

Below variations that can be tried with this 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.

Thank You!!