MARS-GM Multi Headed Recommendation System using Graphical Modeling

Ayushi Agarwal * **Parth Shettiwar** * **Shardul Parab** * **Satvik Mashkaria** *

[1]

Abstract

In past few years, graphical neural networks have been shown effective in performing social recommendations. The common trick is to leverage GNNs to model relation between users and items and to model the overall social structure between the users. Such setup however faces many challenges to integrate the interactions of various elements. Recently GraphRec (5) was introduced to solve this problem which modelled these interactions in a systematic approach. Leveraging this network, we plan to incorporate the latest state of art Attention mechanism, inspired from the Transformer architecture (19), to learn the attention weights between various elements of this graph to ensure embeddings are learnt in a coherent manner. We further introduce adverserial learning by introducing noise in the graph to model real world datasets and avoid overfitting. Additionally, our main focus is also to prove that how can we solve this problem for multiple spaces and show that the approach is easily adaptable to new spaces. We show the performance of our network on the FilmTrust dataset which has a different set of network elements as compared to previous datasets. We compare all these results to the Baseline model Graphrec.

1. Introduction

Recommendation systems are proven to be quite useful in the today's dynamic and competetive world, where they recommend relevant items to a user based on various factors. Such recommendation engines have been used on a large scale in the market in order to ensure better sales, engagement and retention of customers. Collaborative and content based filtering are the two approaches usually used in this domain.

The use of social relations for developing recommendation systems has shown to be effective in past recent years. The intuitive thinking behind this is that users who interact with each other, tend to have some similarity in their in the way they interact with various items, movies and other places where they rate their purchases. With the advent of deep learning in graph structured data, many developments have come in the domain of recommendation systems as usually social interactions among people in a society is modelled by a graph. Graphical Neural Networks (8) have been recently introduced which learn constructive latent vectors from the graph data.

The ideal way to model a recommendation system, leveraging social interactions, is to model a graph between user and items, and another graph to model the user-user relationship. Such dual learning go relations helps to understand the underlying distribution of user to item ideal preferences. However such a modelling, has various inherent challenges. The model has to capture the information from there 2 graphs coherently and in a weighted fashion to output the correct recommendation. At the same time, its important to incorporate users opinions on items too (on users experience of item after using it) in addition to the user item interaction. Another challenge lies in the fact that users tend to interact in a varied and diverse manner, which needs to be differentiated well in the latent space by the network model.

2. Problem Statement and Novelty

Overall, we propose to build upon the existing state of work in this domain, (5), where they introduced GraphRec network was introduced having a separate user and item modelling graph networks, which are then combined for the rating prediction. We aim to solve the following problem:

Let $U = (u_1, u_2, ..., u_n)$ and $V = (v_1, v_2, ..., v_m)$ be the sets of users and items respectively, where n is the number of users, and m is the number of items. We assume that n×m is the user item rating matrix, which is also called the user-item graph. r_{ij} is the rating score, to represent the rating from user u_i to item v_j. The user user social graph T is modelled such that T_{ij} = 1 if user u_j is related to user u_i. be the set of known ratings. . Given the user-item graph R and social graph T, we aim to predict the missing rating

[1] https://github.com/ayu15031/MARS-GM

value in R.

There has was a serious thought process which did go into thinking ways in which we can make take the GraphRec solution as the baseline and find novel ways to extend the given approach. New problems to be tackled are as follows: *The points below contain the new problems that will be tackled as well as their challenges and their reasons to spark interesting research ideas.*

1. **Using Huber loss function and adversarial techniques**: Different loss functions can give different results and can improve the performance of multifold based on tweaking the loss function. Additionally 'adversarial techniques' and their loss functions have also seen a large amount of adoption to solve multiple data mining and machine learning problems. We aim to utilize such techniques to build even more noise immune recommenders with the challenge being that training and getting effective results with such results is highly tricky and time consuming.

2. **Explore performance on multiple datasets**: We also intend to explore performance on different datasets in the social recommendation space. Currently we have picked the Filmtrust dataset along with the Ciao, Epinions provided in the paper.

3. **Expanding the problem to a multi-space problem from a two space problem**: Currently the paper tries to solve only a two space scenario (i.e. user-item and user-user space). However, in the real world, it can be necessary to build and explore relations between more than two spaces to make the recommender more robust and provide even more interesting and insightful recommendations. An example could be say in the for the case of Amazon where it does provide both PrimeVideo and Shopping Services, we can make multiple spaces and relations between users, Shopping items and movies.

4. **Implementing a new GNN social recommendation model**: By using the existing structure explained in the paper in concern as an inspiration, we intend to create our own new model by utilizing new and state of the art attention models.

3. Literature Survey

The recommendation task entails recommending one or more unobserved items to a specific user. As the popularity of e-commerce and social media platforms has grown, recommendation algorithms have become critical tools for many businesses. Different methods have been used for social recommendation. In this project we focus on model-based social recommendation. One of the most common method for this problem is Matrix Factorization. The model states that the preferences of a user are dependent on the people around them. (13) proposed a co-factorization method, which has a shared latent user-feature matrix that is influenced by ratings and social connections. (18) worked on dividing the whole user space intro various clusters and modelled the hetrogeneous relations among them. Some other matrix factorization based methods include (12), (14), (11), (21), and (10) With growing users and items spaces, these methods are incapable of capturing complex and intrinsically non-linear features of social relationships. (17) describes overview regarding social recommendation and some common methods that are used to solve it.

With advancement of the use of deep neural networks in various fields such as Computer Vision, Natural Language Processing etc., the use of neural networks in Recommendation systems [(4),(3)] has just started to capture interests. (4) uses a DNN model to learn non-linear user-user features using social dependency. These methods seems to perform better than the state of the art recommender systems such as (13).

Graph Neural networks have been to prove to learn graph data efficiently and hence they have been used to capture social relations and solve the problem of social recommendation. Through the lens of graph link prediction, (2) investigate matrix completion for recommender systems. They proposed a graph auto-encoder framework for generating latent user and item features on the user-item graph via a differentiable message passing. The majority of existing models assume that social effects from friend users are static, taking the form of constant weights or fixed constraints. To loosen up on this strong assumption paper (20) proposed dual graph attention networks to collaboratively learn representations for two-fold social effects, one of which is modeled by a user-specific attention weight and the other by a dynamic and context-aware attention weight. (9) uses graph neural networks to encode user and item feature spaces by using different graph networks for each of them. These encoded spaces are then converted to latent factors of matrix factorization to complete the rating matrix. Currently they only incorporate the social graph into recommendation, despite the fact that many real-world industries users and items, for example, are linked to a plethora of rich attributes. Similarly, (5) proposed GraphRec, a framework that coherently models two graphs with heterogeneous strengths, and provide a principled approach to jointly capturing interactions and opinions in the user-item graph. This is currently the state of art for Epinions (15) dataset. In (6) the authors proposed GraphRec+ that uses multiple improvements over GraphRec. Inspired by their changes we propose various other changes to increase the efficacy of recommendation systems.

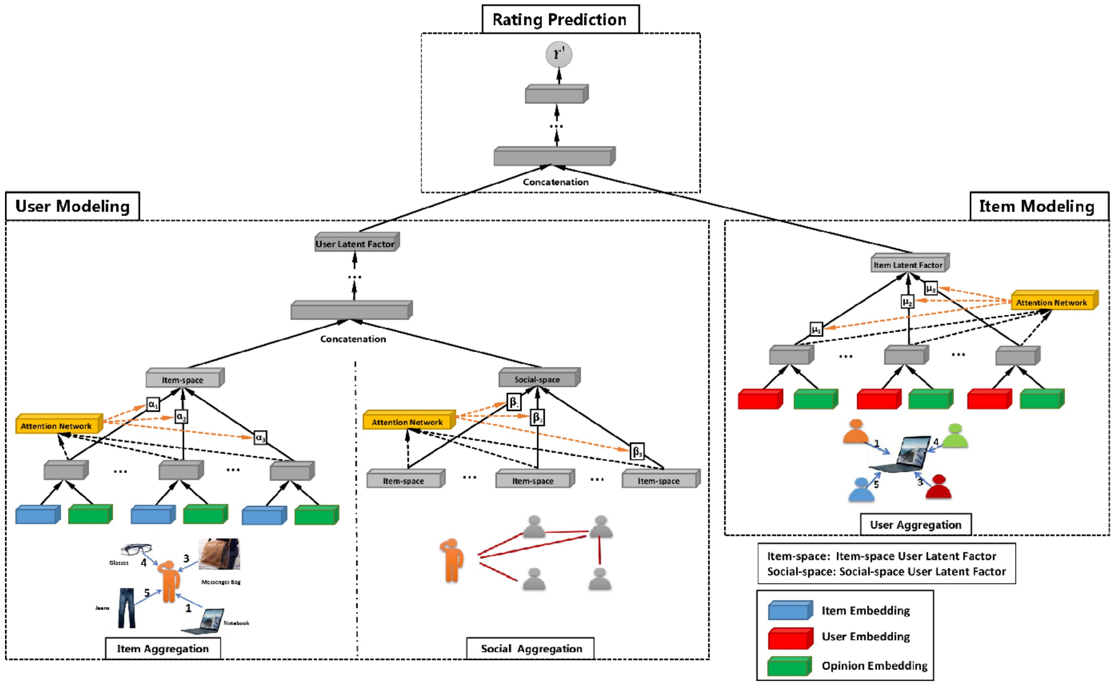


Figure 1. The original state of the art GraphRec Model (5)

4. Datasets

We are considered the following three benchmark datasets for our task.

1. **Epinions**
   This is a who-trust-whom online social network of a a general consumer review site Epinions.com. (15) This dataset contains $75,879$ Nodes and $50,8837$ Edges. Each node is a user and edge indicates whether a user trusts the another user.

2. **Ciao**
   Ciao is a product review website where users may rate and publish product reviews while also forming social relationships. (16) Ciao offers additional contextual information to ratings and social information, such as when ratings are provided, product category information, and review information such as content and helpfulness votes.

3. **FilmTrust**
   The FilmTrust dataset is gathered from a website that combines Semantic Web-based social networks with trust to generate predictive movie recommendations (7). The dataset contains user to movie ratings. The number of these ratings are 35497. The dataset also has 1853 directed trust ratings between 2 users. A thorough data analysis of FilmTrust data is shown in figure 2 that shows the user to item modelling which is a bipartite graph and is highly connected. Thus, showing this dataset can be easily utilized for social recommendation.

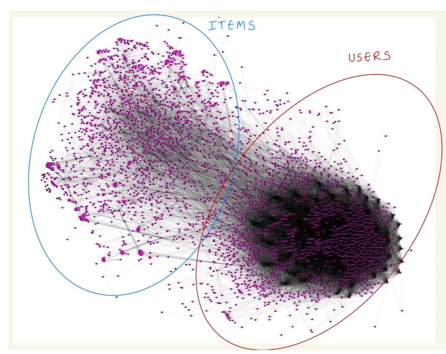


Figure 2. Graph containing user to item modelling i.e an edge between a user and item is present only when a particular user has rated a particular item.

The results of the baseline GraphRec were evaluated on Ciao and Epinions dataset. However, since this dataset was very large we moved to use FilmTrust dataset based on the analysis.

5. Data and Model Pipeline

While developing the data and model pipeline we have paid detailed attention on how the pipeline can be as flexible as possible in order to allow for incremental improvements. The pipeline should be such that it is as modular as possible and different components can simply be added/replaced as an when needed. In this way, we can use the given framework in figure [2] to further research in this domain.

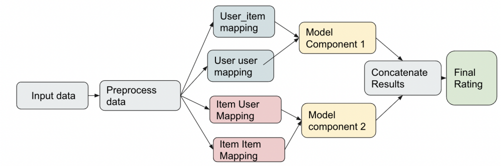


Figure 3. Data and model pipeline for MARS-GM

5.1. Data preprocessing and Mapping creation

Given that we have an input which consist of relations between users, items, ratings, etc we need to preprocess it in order to make it available in the spaces that we want to model our information that needs to be sent to the given model components. In this way, preprocessing leads us to distinct matrices providing us mappings in the various spaces. For example, over here the model to *user-item* and *user-user* mapping (for User modelling) and to *item-user* and *item-item* mapping (for Item Modelling)

5.2. Model Components and Final Rating

All the mappings created and be combined together in different combinations and then be sent to the numerous model components. Here model components can be any simple or complex Machine Learning/Deep Learning model whose main aim is to provided Latent Factor Embeddings of the spaces that we want to take into consideration. Finally these Latent Factor Embeddings are concatenated together, followed by the some additional layers(if needed) to give the final rating prediction.

6. Proposed Model

6.1. Item to item Modelling

As we know in the real world the relations of item is also a factor that accounts for ratings. For example, a person buying an Iphone is more biased towards buying an Ipad or Macbook. Models discussed so far doesn't explicitly incorporate the similarity between items i.e. if a user likes scarf, with high chances it will like muffler as well.

Inspired from GraphRec+ (6) as shown in Figure 5 we added item-item modelling in GraphRec using the following method.

To calculate this item to item similarity, we make the use of previously computed item-user graphs. We say two items are similar if the users who rated those items are similar.

To capture the similarity between two items i_1 and i_2, we form vectors u_1 and u_2 such that j^{th} element of u_1 will be the rating given by j^{th} user to item i_1. If the user has not given a rating for i_1, the entry will be 0.

Then we utilize Pearson's Correlation Coefficient to compute the similarity between different items. So the ratings between two items i_1 and i_2 will be the Pearson coefficient between the user vectors of the two items.

$$Sim(i_1, i_2) = \frac{\sum_j (u_1(j) - \bar{u_1})(u_2(j) - \bar{u_2})}{\sqrt{\sum_j (u_1(j) - \bar{u_1})^2 \sum_j (u_2(j) - \bar{u_2})^2}} \tag{1}$$

For a particular item the top k related items were selected i.e items having the highest similarity and an edge between those items were added to create the item-item graph. For our case the value of k was taken as 30.

The results of item-item modelling for the filmTrust dataset is shown in Figure 6. It can be seen that the graph is highly structured and contains multiple clusters.

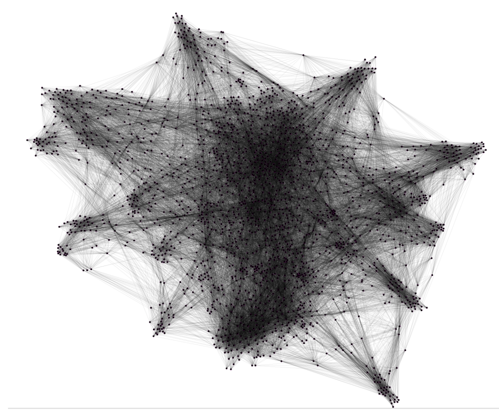


Figure 6. Item-Item Graph

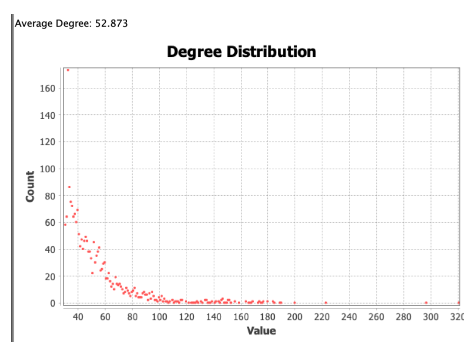


Figure 7. Histogram showing degree distribution of item-item graph

6.2. Multi-Headed Attention

Multi headed attention helps in learning different kind of interdependence in the data heterogeneously. It helps in capturing multiple patterns in all four spaces. Instead of

Training	Metrics	Algorithms								
		PMF	SoRec	SoReg	SocialMF	TrustMF	NeuMF	DeepSoR	GCMC+SN	**GraphRec**
Ciao (60%)	MAE	0.952	0.8489	0.8987	0.8353	0.7681	0.8251	0.7813	0.7697	0.7540
	RMSE	1.1967	1.0738	1.0947	1.0592	1.0543	1.0824	1.0437	1.0221	1.0093
Ciao (80%)	MAE	0.9021	0.8410	0.8611	0.8270	0.7690	0.8062	0.7739	0.7526	0.7387
	RMSE	1.1238	1.0652	1.0848	1.0501	1.0479	1.0617	1.0316	0.9931	0.9794
Epinions (60%)	MAE	1.0211	0.9086	0.9412	0.8965	0.8550	0.9097	0.8520	0.8602	0.8441
	RMSE	1.2739	1.1563	1.1936	1.1410	1.1505	1.1645	1.1135	1.1004	1.0878
Epinions (80%)	MAE	0.9952	0.8961	0.9119	0.8837	0.8410	0.9072	0.8383	0.8590	0.8168
	RMSE	1.2128	1.1437	1.1703	1.1328	1.1395	1.1476	1.0972	1.0711	1.0631

Figure 4. Baseline Results provided by the GraphRec paper (5)

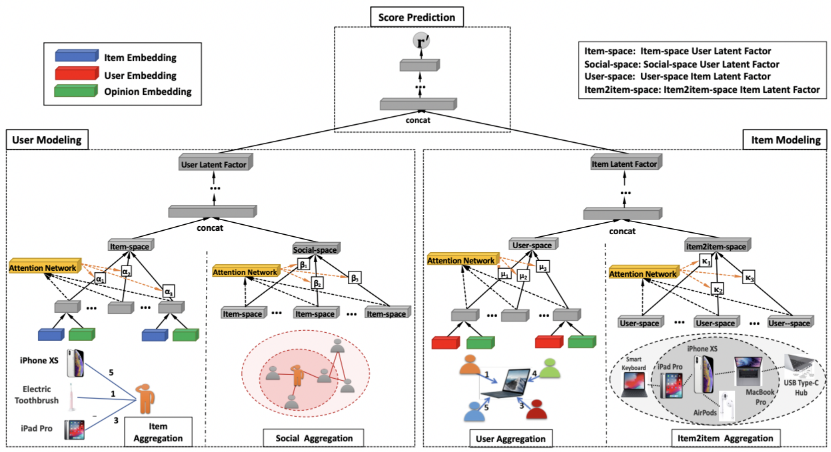


Figure 5. GraphRec+ Architecture taken from paper (6)

training single attention weights, we train multiple attention heads and then combine them to find final attention weights. There are two possible ways we can combine the heads:

1. Take mean across all the heads.
2. Concatenate all the heads and pass them through a fully connected layer to find final weights.

We implemented both the methods and found that the second method works better in practise. We applied multi headed attention in all four spaces: user-user, user-item, item-user, user-user. We use 8 attention heads in all four spaces.

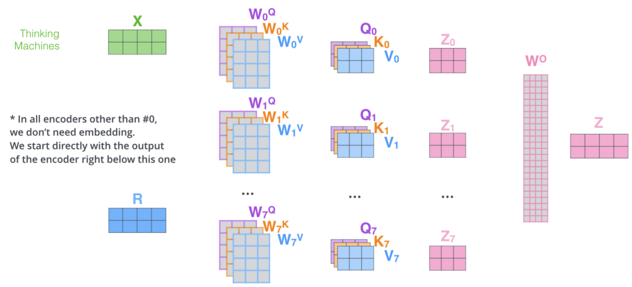


Figure 8. Multi Headed Attention (1)

6.3. Huber Loss

We used Huber Loss, which is an extension over MSE loss. Let's say our prediction is x and the ground truth is p,

$$L_\delta(x) = \begin{cases} (x-p)^2 & \text{if} |x-p| \le \delta \\ \delta(|x-p|-\delta) & \text{otherwise} \end{cases} \tag{2}$$

This loss is designed in such a way that it penalizes the outliers linearly instead of quadratically. This makes sure that loss doesn't get too much affected by outliers, and thus stabilizing the training. Note that the loss is formulated such that it is still continuous and differentiable. We chose $\delta = 1$ after trying out different values and found that using this loss actually improves the performance.

6.4. Adversarial Learning

As part of this ablation study, we tried to observe the effect of noise in ratings given by users. This was done to simulate real world scenario where usually ratings given by users are not exactly equal to what they intended. A gaussian random

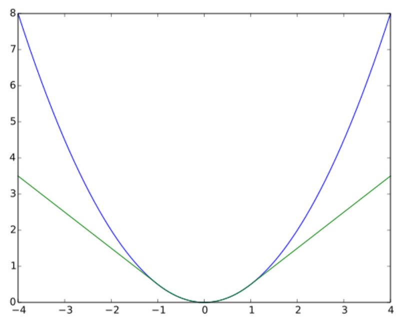


Figure 9. MSE vs Huber Loss

noise with zero mean and variance ranging from = 0.002 to 0.2 was applied and metric scores were observed. The following graph were plotted:

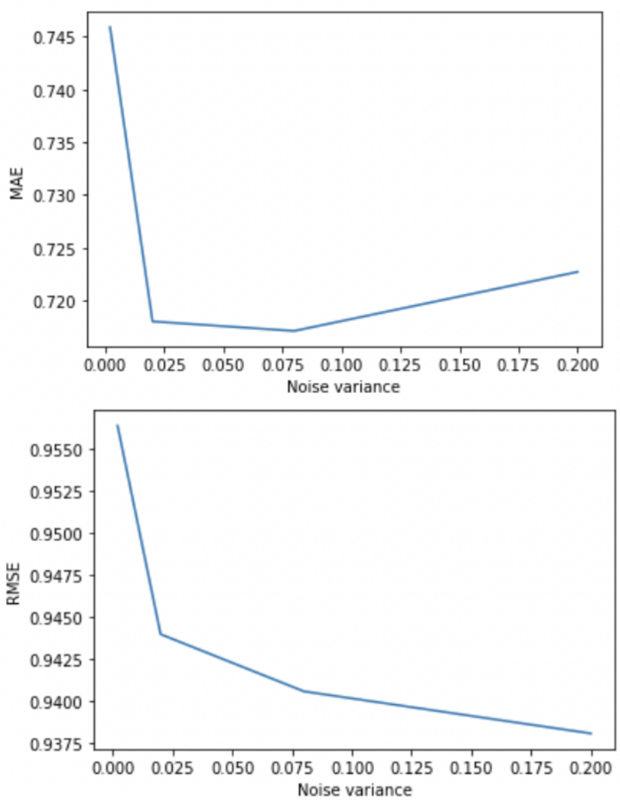


Figure 10. Scores of MAE and RMSE on addition of noise with different variances to link ratings

An interesting observation was made where we see that the accuracy metrics actually improve with increasing noise variance. For very high noise variance, this fails. But at the range of 0.002 to 0.2 which we tried, the scores improve. This suggest that adding such noise actually acts as a regulariser and helps to avoid overfitting of our model. It also, as described helps to model real world case where ratings might have inherent noise in it.

7. Implementation Details

All the experiments were carried out on NVIDIA Tesla K80. Each epoch took about 10 seconds on FilmTrust dataset. Batch size was kept to 256 and embedding dimension for all spaces was kept to be 64. The learning rate was set to 1e-5 which worked best for our case as the dataset was small. Another crucial hyperparameter was truncating length which considered how many ratings to be taken for each user. We kept the `truncating_length` = 15 for all our training purposes. A split of 80:10:10 was made on dataset as training:validation:test sets.

8. Results

We used the FilmTrust dataset to evaluate the results. The dataset consisted of (user, item, rating) matrix and (user, user) trust matrix. This was converted into 4 spaces user-user matrix, user-item matrix, item-user matrix, item-item matrix. These spaces then were used to predict the final ratings.

The results were calculated on Baseline model that is GraphRec model. The dataset was divided into 80% training, 10% validation and 10% testing sets.

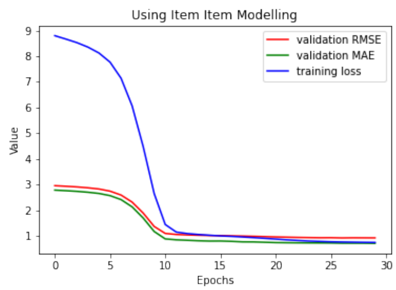
As all the benchmark models use two metrics to evaluate their models that is **Root Mean Square Error (RMSE)** and **Mean Absolute Error (MAE)**, we used these two metrics to evaluate our model.

We calculated the results on the following grounds:

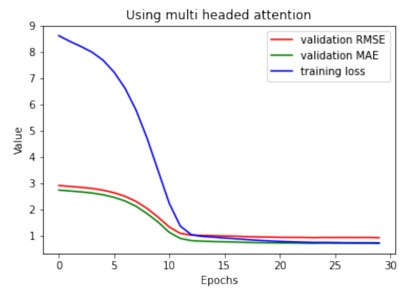
1. Added item-item modelling
2. Added multi-head attention
3. Added item-item modelling and multi-headed attention
4. Added item-item modelling, multi-headed attention and used huber loss

   The training loss and mae, rmse of validation of each of the training are shown in figure 11.

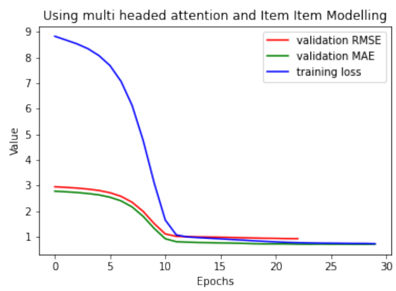
   The RMSE and MAE values of the resulted models have been given in the Table 8. It can be seen that using Item-Item Modeling, Multi Headed Attention individually and together improves the results from the baseline model. It is also seen that when we use Huber Loss instead of tradition Cross-Entropy loss the loss decreases to a great extend as per expected.



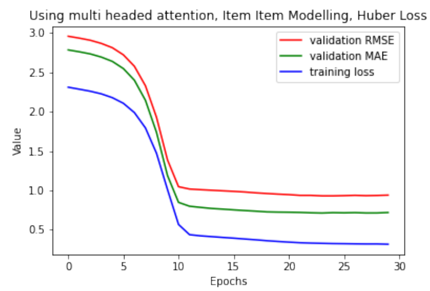
(a) Using item-item modeling



(b) Using Multi-headed attention



(c) (a) + (b)



(d) (c) + Huber Loss

Figure 11. Training losses and RMSE and MAE on Validation

Model	MAE	RMSE
Baseline	0.7153	0.9217
With Item-Item modeling	0.7042	0.9063
Multi-headed attention	0.7116	0.9185
Multi-headed attention + item item modeling	0.7023	0.9045
Huber Loss + Multi-headed attention + item item modeling	**0.7003**	**0.9001**

Table 1. Testing results using different techniques

9. Conclusions and Future Work

We have clearly seen how we can build a simple yet flexible pipeline for social recommendation using graphs. We have also clearly observed as to how we can extend the problem to even more spaces which does directly impact the accuracy of the predictions. Utilizing additional methods such as more powerful loss functions and altering the model component structure does also help in improve the accuracy score of the system

The main *limitation* of our current approach is that we are developing a model which can work really well for static environments. However, in the case of dynamic or online environments, for real world scenarios it is preferable that we build dynamic graphs. Hence, the challenge lies in how one can adopt online data to develop dynamic graph neural networks to produce recommender systems which are even more effective. This will definitely be the major area of focus for the future of social recommendation systems.

10. Division of work

1. Parth - Analysed and filtered FitmTrust dataset. Trained Baseline on FilmTrust dataset. Tested Adversarial Learning. Helped formulating item-item similarity and modeling.

2. Ayushi - Ran Baseline, created item-item graph for FilmTrust, Formulated and coded item2item/extra space modelling and helped training model using item-item modelling

3. Satvik - Added Multi Headed attention. Vizualized datasets using Gephi. Helped formulating item-item similarity and modeling.

4. Shardul - Formulated and coded item2item/extra space modelling. Trained model using item-item modelling and reasearched over various loss functions to use.

References

[1] Multi head attention. https://jalammar.github.io/illustrated-transformer/. Accessed: 2010-09-30.

[2] Rianne van den Berg, Thomas N Kipf, and Max Welling. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263*, 2017.

[3] Shuiguang Deng, Longtao Huang, Guandong Xu, Xindong Wu, and Zhaohui Wu. On deep learning for trust-aware recommendations in social networks. *IEEE Transactions on Neural Networks and Learning Systems*, 28(5):1164–1177, 2017.

[4] Wenqi Fan, Qing Li, and Min Cheng. Deep modeling of social relations for recommendation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[5] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation, 2019.

[6] Wenqi Fan, Yao Ma, Qing Li, Jianping Wang, Guoyong Cai, Jiliang Tang, and Dawei Yin. A graph neural network framework for social recommendations. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[7] Jennifer Golbeck, James A Hendler, et al. Filmtrust: movie recommendations using trust in web-based social networks. In *CCNC*, volume 2006, pages 282–286. Citeseer, 2006.

[8] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks, 2016.

[9] Zhiwei Guo and Heng Wang. A deep graph neural network-based mechanism for social recommendations. *IEEE Transactions on Industrial Informatics*, 17(4):2776–2783, 2020.

[10] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. *CoRR*, abs/1708.05031, 2017.

[11] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, page 135–142, New York, NY, USA, 2010. Association for Computing Machinery.

[12] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. Sorec: Social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, page 931–940, New York, NY, USA, 2008. Association for Computing Machinery.

[13] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296, 2011.

[14] Andriy Mnih and Russ R Salakhutdinov. Probabilistic matrix factorization. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2007.

[15] Matthew Richardson, Rakesh Agrawal, and Pedro Domingos. Trust management for the semantic web. volume 2870, pages 351–368, 10 2003.

[16] Jiliang Tang, Huiji Gao, Huan Liu, and Atish Das Sarma. eTrust: Understanding trust evolution in an online world. In *KDD*, 2012.

[17] Jiliang Tang, Xia Hu, and Huan Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113–1133, 2013.

[18] Jiliang Tang, Suhang Wang, Xia Hu, Dawei Yin, Yingzhou Bi, Yi Chang, and Huan Liu. Recommendation with social dimensions. In *AAAI*, 2016.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[20] Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems. In *The World Wide Web Conference*, pages 2091–2102, 2019.

[21] Bo Yang, Yu Lei, Jiming Liu, and Wenjie Li. Social collaborative filtering by trust. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(8):1633–1647, 2017.