Graph mining project

We are working on recommendation systems to enhance item recommendations for graph data. Our dataset includes two distinct graph structures: a user-user graph, which records interactions and influences among users, and a user-item graph, which records interactions between users and items. The user-item graph is weighted, with edge weights representing the ratings assigned by users to items. Our work focuses specifically on the ciao graph dataset, and more information about this dataset can be found at this link. Overall, our efforts involve working with these two graphs to improve our recommendations.

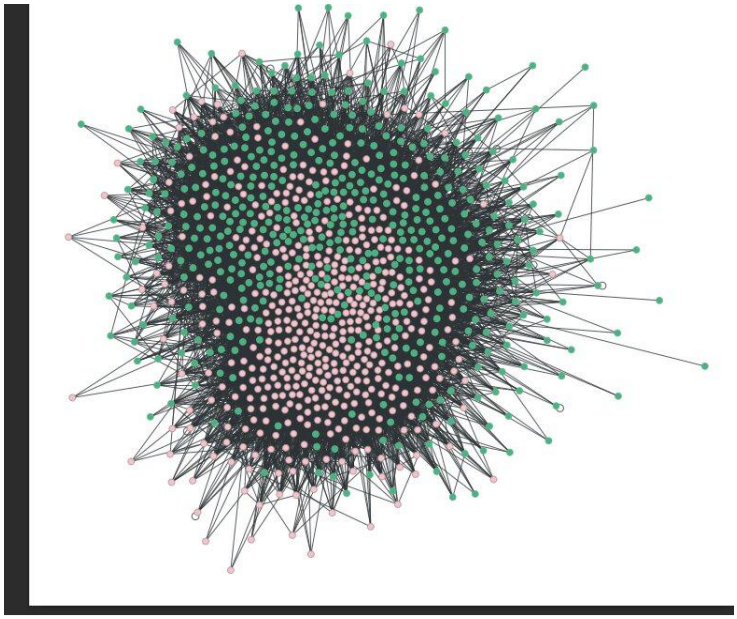
Our work can be divided into three parts. Firstly, to gain a better understanding of the graph data, we visualize and analyze it. Secondly, since our main goal is to develop a recommendation system, we propose two methods for online recommendation. Finally, in order to train a model on the graph data for link prediction, which involves predicting the weight of edges between users and items that are not given, we first preprocess the data. For model training and testing, we split the data and then use the node2vec algorithm to embed the user-user graph and the user-item graph separately. We generate 16 features for each embedding and concatenate them, before feeding them to models such as linear regression and xgboost for link prediction. We then evaluate our results using MAE and RMSE scores.

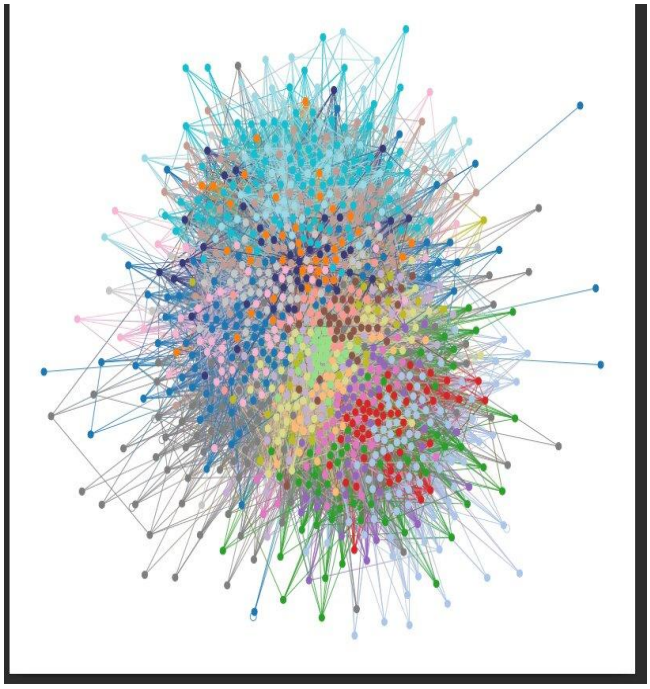
Visualization & Analyst:

Here we give a demonstration of graph properties.
Since our data has 114,173 vertices and 327,721 vertices, it takes a lot of time to display them fully on the page and it also produces an irregular shape.Well, at first we will check the general properties of our graph and then we will show and analyze an example of its display

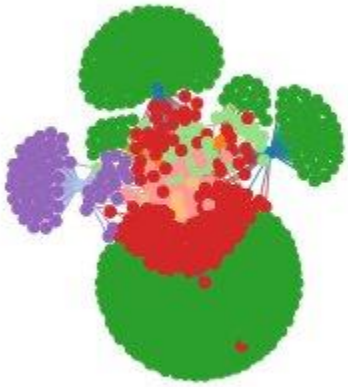
Due to the fact that there is no edge between the goods, logically most of the edge degrees will be between 0 and 50, but there are vertices whose degrees will reach 1500 and also most of our nodes that make up the goods, which are mostly clustering_coefficients They have many, but among them there are products that seem to have been well advertised among friends and have clustering exactly equal to one, and one of these products is item number 18862.

Our graph is made up of a very large component whose number of high nodes reaches 90000 nodes, and higher working nodes play an essential role in our network, and one of the most important high nodes in our network, which handles one third of the network, is node number 273. which is in a way the main gateway of our networNow we will examine a part of the network, which includes the user-user graph As you can see in the picture, for better understanding, the image has been classified and the user's graph is almost uniform and the degree of proximity of all nodes is equal and the graph is connected.





In the product user graph, you can see that the pink color is the symbol of work and the green color is the symbol of the product, which intuitively feels that people close to you bought the same group of products, and the second picture shows that we have three We are a group of goods that have very little in common



As you can see in the classified image, the blue points are users who are close to each other and have bought many products close to each other, and the central ones are very similar and have edges between them and the products. They have a little bite

Online recommendation:

We propose two methods for online recommendation:

1.

This is based on a random walk. We run a personalized random walk on each user then based on that we recommend top k-items to the user.

2.

In this method, we use the k means algorithm to classify the data and suggest the closest one using a level of 10 states.

Link Prediction:

First, we preprocess the ciao dataset. Then, we split it into two sections: training and testing with a test ratio of 20%. Next, we construct two graphs: user-user and user-item graphs. We use node2vec to embed the user-user graph and obtain a 16-length feature set for each user. Similarly, we embed the user-item graph using node2vec and obtain a 16-length feature set for each user and item. For each edge in the user-item graph, we create a 48-length feature by concatenating the user-user embedding feature set and user-item embedding feature set for both user and item (16 + 16 + 16 = 48). Finally, we train various models including linear regression, random forest, and xgboost. Below are the results for each model.

Metric	Linear Reg	Ridge Reg	Lasso Reg	Random Forest	XGBoost
MAE	0.8182	0.8182	0.8207	0.8055	0.7702
RMSE	1.0616	1.0616	1.0631	1.0416	1.0031

The best result we get is from XGBoost. Let's compare it with the models described in the paper for ciao dataset with test and train ratio (80% 20%):

Metric	PMF	SoRec	SoReg	Social MF	TrustMF	NeuMF	DeepSoR	GCMC +SN	GraphRec
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

References:

https://arxiv.org/pdf/1902.07243.pdf
https://paperswithcode.com/dataset/ciao
https://github.com/sm823zw/Recommendation-System-Using-GNNs