Customer-Category Interest Model: A Graph-Based Collaborative Filtering Model with Applications in Finance

Yue Leng[†]
yue.leng@jpmorgan.com
J.P. Morgan Quantitative Research
New York, NY, USA

Evangelia Skiani[*][†]
J.P. Morgan Quantitative Research
New York, NY, USA

William J Peak[†]
J.P. Morgan Quantitative Research
New York, NY, USA

Ewan Mackie[†]
J.P. Morgan Quantitative Research
London, UK

Fuyuan Li[†]
J.P. Morgan Quantitative Research
New York, NY, USA

Thwisha Charvi[†]
J.P. Morgan Quantitative Research
Mumbai, India

Jennifer Law[†]
J.P. Morgan Quantitative Research
London, UK

Kieran Daly[†]
J.P. Morgan Quantitative Research
New York, NY, USA

ABSTRACT

The financial domain naturally contains multiple different types of entities such as stocks, product categories, investment participants, intermediaries, and customers, and interactions between these entities. This paper introduces a Graph-based Collaborative Filtering Category Recommendation (GCFCR) system as a first step in modelling the financial domain as an inter-connected, heterogeneous, dynamic system of nodes and edges. The goal of this paper is to identify customer interest based on the neighborhood of each node and make personalized suggestions or identify relevant content for each customer. Matching relevant products and services to customers is a key foundation of building and maintaining strong customer relationships, facilitating more personalized marketing which can ultimately result in increased customer activity, trust, and revenue. In this paper, we run a set of experiments to compare different recommendation techniques, concluding that the proposed GCFCR approach outperforms in this real-life application.

CCS CONCEPTS

• **Information systems → Recommender systems**; • **Collaborative filtering**;

KEYWORDS

Graph Neural Network, Recommendation Systems, Machine Learning in Finance, Marketing

[*]Corresponding author
[†]Opinions expressed in this paper are those of the authors, and do not necessarily reflect the view of JP Morgan

ACM Reference Format:
Yue Leng, Evangelia Skiani, William J Peak, Ewan Mackie, Fuyuan Li, Thwisha Charvi, Jennifer Law, and Kieran Daly. 2022. Customer-Category Interest Model: A Graph-Based Collaborative Filtering Model with Applications in Finance. In *3rd ACM International Conference on AI in Finance (ICAIF '22), November 2–4, 2022, New York, NY, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3533271.3561757

1 INTRODUCTION

Machine learning and data science models are researched and deployed in a wide variety of fields, and Finance is no exception. Initially, applications were identified in the traditional area of stock market investments, where there is a growing need to analyze large-scale data for systematic algorithmic trading; an area, which requires fast processing of high frequency streams of data for investment decisions or strategy selection. Recommendation engines are a subclass of machine learning that has thus far attracted relatively less attention within Finance. However, recommendation systems have been extensively used by Technology firms. For example, Netflix [4], where movies are being recommended to users, or e-commerce firms such as Amazon [25] where shoppers are presented with a pool of highly relevant items the goal is often to generate efficiencies in the user experience and income across multiple products.

In the majority of cases, the use of state of the art machine learning techniques have enabled firms to stand out significantly from their competitors. In a highly competitive environment such as Finance, where banks, actively compete to provide the best services to their customers, "knowing your customers" lies at the core of each financial institution. In this paper, we are focusing on the Marketing organization which bears most of the customer-facing weight and can greatly benefit from the deployment of machine learning techniques.

The Marketing department traditionally deals with increasing profits by efficiently selling the bank's products and services to customers. They usually work with the product teams to market products and services which appeal to their customers, increasing loyalty from existing customers whilst attracting new business. An experienced Marketing person would know what their customers are interested in, which products they would like, what information they would likely consume and which services they would favor.

There remain many challenges in "knowing your customer". This is true, especially when it comes to smaller customers, less experienced salesforce, newly opened accounts, outdated information or even information which is difficult to collect and aggregate.For example, information that could be stored in many databases or circulated via e-mail or spreadsheets.

We identify and expose cases where the Marketing operational model could be optimized via the analysis of large datasets, especially in cases where there is a long history of data collection. Customer communications can be significant indicators of what each customer is interested in. This information, which in the past would sit within the Marketing department, Sales spreadsheets or e-mail chains, is transformed into a well-defined dataset. Then, the dataset could be fed into a sophisticated model to identify relationships and recommend interest tags. In this context, we would like to infer what a customer would be interested in and make recommendations. In practice, Marketing team will typically combine their past experience and the recommendations made by the support systems over a period of time to make informed decisions.

In this paper, we use a graph neural network that is built based on the interaction data between customers and product categories. Other types of nodes used herein can be viewed as proxies, especially when interaction data is missing. By building edges, i.e., connections between customers and the corresponding product categories, we construct a proximity area around each customer and capture similarities between similar nodes as well as direct relationships with the other types of nodes throughout the network. We will provide more details on the construction of the graph in Section III. The main contributions of our paper are as follows:

- We propose a Graph-based Collaborative Filtering Category Recommendation (GCFCR) framework that is able to personalize product category recommendations to customers.
- Our framework exploits the high-order connectivity from customer interactions into the embedding function. This is to encode collaborative signals in the interaction graph structure.
- We empirically validate our framework by analyzing real-world data that records historical interactions of customers and product categories. The proposed framework is compared with multiple existing methods to demonstrate efficiency and effectiveness of injecting user-item graphs into the embedding learning process.

The proposed framework could be beneficial to both new and existing customers. It will automatically match customer interests to create a list of the most relevant product categories within the graph. The customer and product categories network is represented by a bipartite graph as shown in Figure 1 , where the nodes on the left represent the customers and nodes on the right represent product categories. Interaction between customers and certain categories are represented by the edges in the graph. Specifically, all interactions may be "tagged" as involving a product category that the customer is interested in. From this bipartite graph, multiplication of matrices can be operated directly on the adjacent matrix corresponding to the bipartite graph to get high order neighbors. For example, two customers are 2-hop neighbors if they have interest in a same product category in the network. Extensive experiments have been conducted using a long history of logged interaction data. The experimental results show that our GCFCR framework outperforms multiple baseline models including traditional collaborative filtering, logistic regression, random forest, and gradient boosting. In addition, our framework can visualize embeddings of customers and product categories to provide insights on neighborhood relationships. This paper is organized as follows: Section II presents related work and motivation, while Section III presents the methodology. In Section IV, we discuss the setup and construction of the experiments. Then, Section V demonstrates the main results. Finally, we conclude with Section VI.

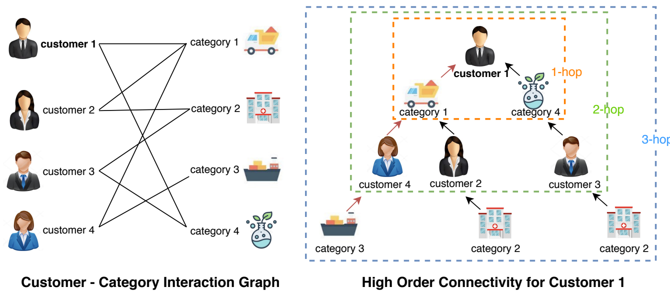


Figure 1: Example of customer-category interaction graph and high-order connectivity for customer 1 (e.g. c_1 is the target person to receive potential recommendations)

2 RELATED WORK

In recent years, recommender systems [5] have become an indispensable function of many applications [41] . As mentioned above this is especially true for "Big Tech", but increasingly true for finance in areas such as fraud detection. The rapid growth of data and information recorded about products (e.g., prices, demand) and consumer behavior (e.g., consumer behaviors, preferences, profiles) enable an expanding scope for improving these systems.

Generally speaking, recommendation systems generate personalized options to individual users by analyzing a large set of possible options with the objective of reducing the complexity of the large amount of information [7]. In this connection, they serve as an information filtering technique that provide appropriate highlighting of products or services to interested users based on historical information concerning the user and the product. In finance, complicated user-user relationships exist as different customers may operate in different regions, and still have the same objectives. In addition, financial data is complex and difficult to model, for example, the variety of modelling techniques and stylized facts present in [8]. In particular, financial time series exhibit such stylized facts as stochastic correlations, heteroskedasticity, fat tails, and in addition may need to be gathered from multiple disparate, noisy sources. Customer preferences, such as risk aversion or market sentiment, is also notoriously difficult to estimate with precision but has been attempted through investor surveys [29].

Collaborative filtering (CF) is a data mining technique adopted by many modern recommendation systems for its ability to capture user-user similarities. It provides recommendations by assuming similar users will have similar preferences for items. CF models have shown high prediction quality both in academic research and industrial applications [24]. To establish recommendations, CF models usually factorize users and items to reconstruct historical interactions among them [9]. An item is then recommended to a user if it has proven popular with similar users.

It is then left to the recommender-developer to define how to model (i) the historical interactions and (ii) the appropriate metric of "popularity". It is often useful to store underlying data in a graph structure with users and items as nodes and relationship-strengths defined by their connecting edges.

The method applied in this paper is the bipartite graph representation where there are two distinct node types (e.g., consumers and product categories). In addition, a node of type 1 may only be connected to nodes of type 2 and vice versa. Bipartite graphs make the mathematics of recommender systems easier [36, 38]. Bipartite graphs are also easier to represent as adjacency matrices, which can make computation easier.

In the recent literature researchers have developed algorithms based on Knowledge Graphs (KGs) as an information resource for CF-based recommendation systems [18, 26]. KGs may help CF methods with the so-called "cold start" problem where the technique may make poor recommendations until a sufficient history of user-interactions has been built. Although we don't not use a KG here, this may be an avenue for future research.

Another benefit of the graph technology is that graph neural networks (GNNs) can be developed as end-to-end learning structure [10–12, 15, 23]. Because of high interpretability and performance, GNN-based recommendation systems have been developed in various application domains: Ying et al. [39] built GNN system using bipartite graph for recommendation on Pinterest. Fan et al.[13] and Guo et al.[18] applied GNNs to model social relations (e.g., interactions among users and their associated opinions) for social recommendations (e.g., predicting user's opinion or rating scores on social information). Wang et al.[34, 35] proposed GNN-based recommendation models, which were applied to real-world data and shown to be effective in a variety of scenarios including movie, book and news recommendations. Compared to traditional CF techniques, one of the main advantages of GNN is the ability to exploit higher-order information of KGs through machine-learned feature embedding.

Despite the success of GNN-based recommendation systems in other domains, their applications in financial engineering are not well explored. The focus of this paper is to throw more light on the effectiveness of such systems for recommendations in the Financial domain. Our goal is to develop an advanced graph-based CF framework that captures the complex interrelationship among customers and products through a product category classifier.

3 METHODOLOGY

The goal of this study is to develop a recommendation framework to predict a customer's interest in a particular product category within the next 3 months, based on the history of complex interrelationships between customers and product categories. This contributes to the Marketing objectives of knowing their customers and automates an important part of their investigatory process. Our proposed GCFCR adopted the Neural Graph Collaborative Filtering (NGCF) to describe the user-item (or customer-category in this present paper) graph structure by propagating embeddings. This enables us to model high-order connectivity within the graph and effectively inject the collaborative signal into the embedding process [37]. Instead of expanding the interaction graph as a tree, the NGCF structure propagates embeddings recursively on the graph. This is achieved by the specifically designed embedding propagation layer, which refines a customer's (or a product category's) embedding by aggregating the embeddings of the interacted categories (or customers). Then, the collaborative signal in high-order connectivity could be captured by stacking multiple embedding propagation layers. For example, let c_m and i_n denote customer m and category n, respectively. As shown in Figure 1 (right), stacking multiple layers enable the framework to capture behavior similarities (e.g., $c_1 \leftarrow i_1 \leftarrow c_2$) and potential recommendations (e.g., $c_1 \leftarrow i_1 \leftarrow c_2 \leftarrow i_2$) with trainable weights reflecting the recommendation priorities. Figure 2 shows the general framework of GCFCR:

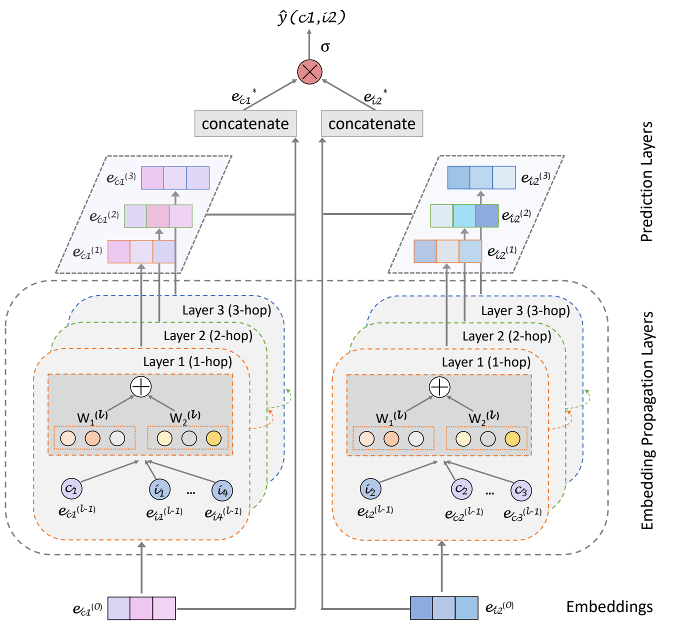


Figure 2: Illustration of GCFCR framework (the arrows present the flow of information). The representations of customer 1 (left) and category 2 (right) are refined with multiple embedding propagation layers, whose outputs are concatenated to make the final prediction.

3.1 Initial Embedding

The embedding vectors $e_{c_m} \epsilon \mathbb{R}^d$ and $e_{i_n} \epsilon \mathbb{R}^d$ with m = 1...M and n = 1...N describe customer c_m and product category i_n, respectively, where d denotes the embedding size.

$$E = [e_{c_1}, e_{c_2}, \ldots, e_{c_M}, \quad e_{i_1}, e_{i_2}, \ldots, e_{i_N}], \tag{1}$$

This embedding table (using normal initializer) serves as an initial state for customer embeddings and category embeddings, which will be optimized during training. In our GCFCR framework, the embeddings are refined by propagating them on the customer-category interaction graph, which results in better recommendations.

3.2 Neighbors

Intuitively, the interacted categories, which are the directly linked (one-hop) neighbors provide direct evidence on a customer's preference. Thus, these preferred categories can be treated as a customer's features and used to measure the collaborative similarity of two customers. Motivated by related research such as [6, 32], where only one-hop neighbors are aggregated, the GCFCR framework also injects high order connectivity relationships (indirectly linked neighbors) via multiple embedding propagation layers to refine the embeddings.

3.3 Message Passing

The operation consists of sending collaborative filtering information along the graph structure from one node to another and is called message passing. This mechanism, introduced by Gilmer et al. [16], acts as the convolution operations in Euclidean spaces, whereby it allows for the spread of information to neighbors and refines the embeddings of nodes. Usually, the message passing has three steps: message construction, message aggregation, and message update [28].

Message Construction: the message is transformed into an embedding. For a connected customer-category pair (*c, i*), let $m_{c\leftarrow i}$ denotes the message from category i to customer c, and $m_{c\leftarrow c}$ denotes the self-connection of c, then:

$$m_{c\leftarrow i} = p_{ui}(W_1 e_i + W_2(e_i \odot e_c)) \tag{2}$$

Where p_{ui} is the discount factor, reflecting how much the historical product category contributes to the customer preference. $W_1, W_2 \in \mathbb{R}^{d' \times d}$ are the trainable weight matrices, and $e_i \odot e_c$ is the element-wise product of e_i and e_c. This setting helps to pass more messages from the similar items, and thus increases the model representation ability and boosts the recommendation performance.

Message Aggregation: this step disseminates the message to every neighbor of the node. Here, one or more embedding propagation layers are stacked to explore different order connectivity information, which are crucial to encode the collaborative signal to estimate the relevance score between a customer and a product category. The recursively aggregated representation of customer *c* after the *l* step (stacking *l* embedding propagation layers) in the function is defined as:

$$e_c^{(l)} = LeakyReLU(m_{c\leftarrow c}^{(l)} + \sum_{i\in Neighbor_c} m_{c\leftarrow i}^{(l)}) \tag{3}$$

In addition to the messages propagated from its connected neighbors $Neighbor_c$, the self-connection of *c* is also considered. The messages being propagated are defined as follows:

$$\begin{cases} m_{c\leftarrow i}^{(l)} = p_{ui}(W_1^{(l)} e_i^{(l-1)} + W_2^{(l)}(e_i^{(l-1)} \odot e_c^{(l-1)}) \\ m_{c\leftarrow c}^{(l)} = W_1^{(l)} e_c^{(l-1)} \end{cases} \tag{4}$$

where $W_1^{(l)}, W_2^{(l)} \in \mathbb{R}^{d_l \times d_{l-1}}$ are the trainable transformation matrices, and d_l is the transformation size; $e_i^{(l-1)}$ is the category representation generated from the previous message passing steps, holding the information from its (*l-1*)-hop neighbors.

Message Update: Finally, the update step will look at every received message and refine the representation of the node accordingly.

As Figure 1 (right) shows, the collaborative signal such as $c_1 \leftarrow i_1 \leftarrow c_4 \leftarrow i_3$ can be captured in the embedding propagation process. Furthermore, the message from i_3 (category 3) is explicitly encoded in $e_{c_1}^{(3)}$ (customer 1, indicated by the red arrows). As such, stacking multiple embedding propagation layers seamlessly injects collaborative signal into the representation learning process.

3.4 Prediction

In this step, the refined embeddings from different propagation layers are aggregated and then outputs the affinity probability of a customer-category pair. After propagating with L layers, we obtain multiple representations for customer *c*, namely $\{ e_c^{(1)}, e_c^{(2)}, \ldots, e_c^{(L)}\}$. Here, the representations obtained from different layers emphasize the messages passed over different connections. The final embedding of a customer is by concatenating them together; The same operation is applied on categories as well, i.e. concatenating the category representations $\{e_i^{(1)}, e_i^{(2)}, \ldots, e_i^{(L)}\}$ learned by different layers to get the final category embedding.

$$e_c^* = e_c^{(0)} || \cdots || e_c^{(L)}; \quad e_i^* = e_i^{(0)} || \cdots || e_i^{(L)} \tag{5}$$

Finally, we conduct the inner product to estimate the customer's preference towards the target category and then apply a sigmoid function to obtain the predicted interaction probability.

$$\hat{y}_{GCFCR}(c, i) = \sigma(e_c^{*T} e_i^*) \tag{6}$$

3.5 Optimization

We used the pairwise Bayesian personalized ranking (BPR) loss [30] and binary cross entropy (BCE) loss to learn the model parameters. For BPR loss, it assumes higher score for observed customer-category interactions (*c, i*) comparing to unobserved ones (*c, j*).

$$Loss_{bpr} = \sum_{(c,i,j)} - ln\sigma(\hat{y}_{ci} - \hat{y}_{cj}) + \lambda\|\theta\|_2^2 \tag{7}$$

$$Loss_{bce} = -\sum_{(c,i)} y_{ci} ln(\hat{y}_{ci}) + (1 - y_{ci}) ln(1 - \hat{y}_{ci}) \tag{8}$$

where θ are trainable parameters in the model, and λ controls the L2 regularization strength to prevent overfitting.

Algorithm 1 summarizes the overall workflow of the proposed recommendation framework, which contains 3 main phases.

4 EXPERIMENT SETTINGS

4.1 Data

To evaluate the effectiveness of the proposed framework, our experiments are performed using a real-world financial dataset over

Algorithm 1 Overall workflow of GCFCR

```
Input: Customer-category interaction history
Output: Interaction probability for customer-category pair

Phase 1 – Preprocessing
1: Dataset preprocessing to extract the different customers and product categories.
2: Prepare adjacent matrix between customers and categories. Assign different weights to different interaction periods (e.g., 1-month ago, 2-month ago, etc.) and interaction types (see Table 2). The weights combine the coefficient learned via a simple shallow neural network (trained separately, outside GCFCR framework) and domain knowledge.

Phase 2 – Training
3: Sample batches of customers/categories from buffer.
4: Message passing to update the embeddings for customers/categories.
5: Compute the loss based on inner product between customer and category.
6: Backpropagate the loss, update the weights and embeddings.
7: Repeat the above steps until stopping criteria satisfied.

Phase 3 – Inference
8: Use embeddings learned from Phase 2 as latent features of customers and product categories.
9: Predict the interaction probability for next three month based on these learned latent features.
```

Table 1: Example of log data

Date	Customer	Category	Product	Event Type
2022-01-01	Alice	XY	XY-001	Meet
2022-01-01	Bob	YZ	YZ-001	Call

a period of 2 years collected at J.P.Morgan. Once we see a (customer,category) pair in a log, the interaction frequency between them will be increased by 1 (Table 1).

Most of the customers typically focus on specific product categories based on their own preferences, therefore, the density for the data is about 9%. We use a rolling period to generate training and testing datasets. The time difference between the training period and the testing period are 6 months For example, in the training dataset, interactions from the Jan 1st - Dec 31st in year 1 are used as features, and the interactions in Jan 1st -March 31st of year 2 are the ground truth labels, which are used to train the model. In the testing dataset, the features come from June 1st in year 1 to May 30th in year 2, and the ground truth labels come from the interaction in June 1st - Aug 31st in year 2. Table 1 presents the summarized statistics of the data.

In general, there are eight types of interactions between customers and product categories. Out of the total 16,642 unique customers in the combined training and testing set, 14,953 customers have interaction records in both training and testing sets, which is 92.9% of all the customers in training data and 96.5% in testing data. Besides that, there are 1,148 customers only present in the training set and 542 customers only present in the testing set.

Table 2: Overall statistics of training and testing datasets

Dataset	# of customers	# of product categories	# of interactions	Density
Training	16,101	35	254,432	0.0901
Testing	15,495	35	239,183	0.0905

The number of interactions per year per customer (with different product categories) and the number of interactions per year per product category (with different customers) are summarized in Table 3.

Table 3: Statistics of interactions

(a) Number of interactions per year per customer

Dataset	Mean	Std.	25th percentile	50th percentile	75th percentile
Training	15.80	97.78	2	6	15
Testing	15.43	80.21	2	6	15

(b) Number of interactions per year per product category

Dataset	Mean	Std.	25th percentile	50th percentile	75th percentile
Training	7,269.49	6,017.00	3,695	5,706	9,979
Testing	6,833.80	5,651.83	3,374	5,396	9,164

4.2 Evaluation metrics

After training the models, we use the past 12 month data to predict the probabilities of customers having interactions with categories. The labels are binary. That is, if a customer will have an interaction with a category in the next 3 months, then the label will be 1, otherwise it will be 0. We see this as a link prediction problem, where customer and category are two types of nodes in a graph and they are linked based on their direct associations. To evaluate the performance of the proposed models, we use Average Precision Score, Precision, and Recall, which are considered to be the most popular metrics for evaluating information retrieval systems [19, 21]. Precision indicates the ratio of correctly predicted interactions to the total predicted interactions and Recall indicates the ratio of correctly predicted interactions to total interactions. In addition, we need to separate the predicted probabilities into binary classes with a threshold (i.e., probabilities greater than or equal to the threshold is associated with having interactions). In this study, the threshold is set to be 0.5 when calculating Precision and Recall. Furthermore, the Average Precision Score is used as the main evaluation metric as it considers all such possible thresholds (which is similar to the AUC for ROC curve). It is a useful metric to compare how well models are ordering the predictions, without considering any specific decision threshold. Other commonly used metrics such as precision@k or recall@k are not used in our study because the use case for GCFCR is not recommending the top k relevant product categories.

4.3 Baseline models

To evaluate the performance, we compare the proposed Graph-based Collaborative Filtering model with multiple baseline models that are commonly applied in recommendation systems:

- **Rule based model**: This is a naive model with a simple rule - if a customer has any interaction with a product category within the past 12 months, then the predicted label for next 3 months will be 1. Otherwise, the predicted label will be 0.
- **Collaborative filtering based recommendation system**: This method [24, 31] is commonly adopted in recommendation systems. Through singular value decomposition (SVD) technique [40], it analyzes customer-customer similarity and category-category similarity. Then the latent customer and category features from SVD are used as input for gradient boosting model for prediction [27].
- **Logistic regression**: In general, a logistic regression model [20] is used to predict the probabilities of the possible outcomes of a dependent variable, given a set of independent variables. In this study, the past 12 months interaction record of a (customer, category) pair is described as a vector (e.g., [1,0,0,0,0,0,0,0,0,0,0,2] indicates the customer has 1 interaction in the 1st month and 2 interactions in the 12th month with the category) to predict the probability for the customer to have interactions with the product category in the next 3 months.
- **Random Forest**: Random forest [33] is a popular bagging algorithm that helps reduce overfitting issues by building multiple decision trees. Each tree is trained slightly differently on a random subset of the entire data given to each tree. During the training at each node of the tree, features used to split the data are randomized. After training individual trees, the final prediction is made based on the average taken across all the trees in the forest. Besides helping to reduce overfitting, the random forest algorithm generally provides a higher level of prediction accuracy compared to the decision tree algorithm [3]. In the present study, the features are the same as what is used for the logistic regression.
- **Gradient boosting**: Gradient boosting models [27] construct additive regression models by sequentially fitting base learners using the residuals at each iteration. The final prediction is made by ensembling the base learners. Previous research has shown that both the approximation accuracy and execution speed of gradient boosting can be substantially improved by incorporating randomization into the procedure [14]. In the present study, the features are the same as what is used for the logistic regression.

4.4 Model selection and Parameter setting for GCFCR

The proposed Graph-based Collaborative Filtering Category Recommendation model (GCFCR) is implemented in Tensorflow [1]. We explored embedding sizes for customer and category as (16, 32, 64). Similar to NGCF [37], the model is optimized with Adam optimizer [22]. The batch size is fixed to be 1024. In addition, we apply a grid search for hyperparameters with: a) learning rate tuned among (0.0001, 0.0005, 0.001, 0.005, 0.01); b) coefficient of L2 normalization tuned among (10^{-5}, 10^{-4}, …, 1, 10); c) the dropout ratio among (0.0, 0.1, 0.2); d) loss function among Bayesian personalized ranking (bpr) [30], Binary cross entropy (bce), Bayesian personalized ranking + Binary cross entropy (bpr+bce). Xavier initializer [17] is applied to initialize the model parameters. Furthermore, early stopping strategy is performed. The max depth of layer L is set to be 3 in order to model the CF signal encoded in the third-order connectivity.

For each observed customer-category interaction, we treat it as a positive instance, and then conduct the negative sampling strategy to pair it with one negative item that the customer does not interact with before. While random sampling the positive samples for a customer, the chance of an category being sampled is set to be proportional to the value with this customer in the adjacent matrix. Note that the cells in the adjacent matrix represent the historical interaction frequency score between customers and categories. The values of the cells are calculated as the weighted sum of a set of interaction frequency counts representing different types of events and interactions for every month (some events are more important than other events: e.g., More recent interactions are associated with higher scores compared to interactions from long time ago, etc.). The final weights for different events combine the learned coefficient from a shallow neural network and marketing domain knowledge.

Table 4: Overall performance comparison

Model	Average precision score	Precision	Recall
GCFCR-2	**0.4621**	0.5804	0.3723
Gradient boosting	0.4275	0.5815	0.3599
Logistic regression	0.4140	0.3152	0.6271
Random forest	0.3703	0.4769	0.3932
Collaborative filtering	0.3660	0.3712	0.5048
Rule based model	0.3626	0.2491	0.4483

Note: preferred model in **bold**.

Table 5: Results of GCFCR variate versions

Model	Average precision score	Precision	Recall
GCFCR-1	0.4529	0.5242	0.4675
GCFCR-2	**0.4621**	0.5804	0.3723
GCFCR-3	0.4583	0.5827	0.3616
GCFCR-2 bpr	0.4549	0.5765	0.3729
GCFCR-2 bce	0.4534	0.5845	0.3698

Note: GCFCR-1/2/3 gather information from first/second/third-order neighborhood layers with bpr + bce loss and embedding size 32

5 RESULTS

Performance of the proposed GCFCR model has been evaluated using the dataset described in Section IV. Table 4 shows the experiment results. The best model is GCFCR-2 (i.e., the GCFCR variate version that gather information from second-order neighborhood layers) with hyperparameters set as: embedding size 32, learning rate = 0.001; coefficient of L2 normalization = 10^{-4}; dropout ratio = 0.1; and loss function = bpr+bce). The results of ablation study is shown in Table 5. It can be observed that the GCFCR-2 outperforms all the baseline models by a large margin (more than 3.4% in average precision score).

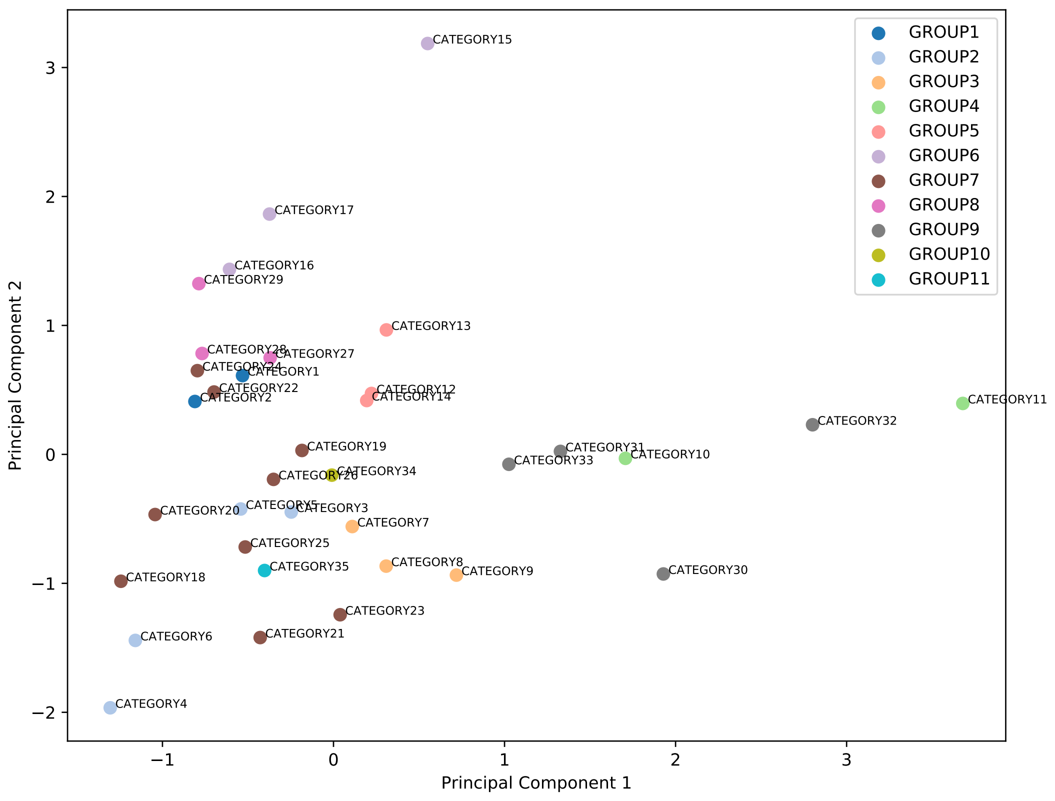
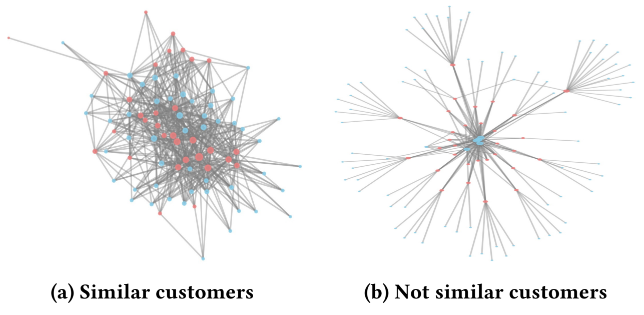


Figure 3: Visualization of category embeddings learned from GCFCR in 2D

To better visualize the relative position of each product category, we project the 32 dimension embedding into 2 dimension with principal component analysis (PCA) [2]. In Figure 3, each dot is a product category. There are 35 dots in total. The color of dot indicate the higher level product group for each category. If categories have same dot color, means they belongs to same higher level groups. We can notice that the relative position of categories matches well with the intuitive understanding of their relation to other categories.

It could be difficult to clearly visualize all customer embedding as shown in Figure 3 due to the large number of customers. We find a customer with relative high interactions count as a representative and plot top 100 similar customers (Figure 4 (a)) and least 100 similar customers (Figure 4(b)). In 2 mode networks, the red dots refer to categories and blue dots refer to customers. The size of dot indicates the degree of the node. It can be observed that similar customers tend to form a big cluster and dissimilar customers tend to form tree shaped networks.

Figure 4: Visualization of the similar (a) and not similar (b) customers with 2-mode network



(a) Similar customers **(b) Not similar customers**

6 CONCLUSION

We have proposed a Graph-based Collaborative Filtering Category Recommendation (GCFCR) framework which can help to capture complex interrelationship among customers and product categories, and therefore, make more effective and personalized recommendations compared to traditional methods. This is because of its capability of modeling the interactions within high order neighbors in the graph. We conduct experiments using a real-world financial dataset collected at J.P.Morgan. The results show that the proposed GCFCR outperforms other methods which do not take the graph structure into account. In addition, variate versions of GCFCR are developed to investigate the impacts of different order of neighborhood layers on product category recommendation. There are several future research directions arising from this work that could be pursued. Currently, we formulate the interaction data as bipartite graph (i.e., customer and category). Future research could include more types of nodes such as sales, competitors, etc. to develop heterogeneous graph. In addition, the proposed recommendation framework can be fused with other advancement in machine learning domain including attention mechanism to better learn the importance of each neighbor, recurrent network to construct the adjacent matrix that captures changes in user's interest over time.

ACKNOWLEDGMENTS

Opinions and estimates constitute our judgement as of the date of this Material, are for informational purposes only and are subject to change without notice. This Material is not the product of J.P. Morgan's Research Department and therefore, has not been prepared in accordance with legal requirements to promote the independence of research, including but not limited to, the prohibition on the dealing ahead of the dissemination of investment research. This Material is not intended as research, a recommendation, advice, offer or solicitation for the purchase or sale of any financial product or service, or to be used in any way for evaluating the merits of participating in any transaction. It is not a research report and is not intended as such. Past performance is not indicative of future results. Please consult your own advisors regarding legal, tax, accounting or any other aspects including suitability implications for your particular circumstances. J.P. Morgan disclaims any responsibility or liability whatsoever for the quality, accuracy or completeness of the information herein, and for any reliance on, or use of this material in any way. © 2022 JPMorgan Chase & Co., all rights reserved.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. {TensorFlow}: a system for {Large-Scale} machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*. 265–283.

[2] Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics* 2, 4 (2010), 433–459.

[3] Naomi Altman and Martin Krzywinski. 2017. Ensemble methods: bagging and random forests. *Nature Methods* 14, 10 (2017), 933–935.

[4] James Bennett, Stan Lanning, et al. 2007. The netflix prize. In *Proceedings of KDD cup and workshop*, Vol. 2007. New York, 35.

[5] Sarah Bouraga, Ivan Jureta, Stéphane Faulkner, and Caroline Herssens. 2014. Knowledge-based recommendation systems: A survey. *International Journal of Intelligent Information Technologies (IJIIT)* 10, 2 (2014), 1–19.

[6] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. 2013. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203* (2013).

[7] Robin Burke. 2002. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction* 12, 4 (2002), 331–370.

[8] John Y Campbell, Andrew W Lo, A Craig MacKinlay, and Robert F Whitelaw. 1998. The econometrics of financial markets. *Macroeconomic Dynamics* 2, 4 (1998), 559–562.

[9] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *The world wide web conference*. 151–161.

[10] Sikai Chen, Jiqian Dong, Paul Ha, Yujie Li, and Samuel Labi. 2021. Graph neural network and reinforcement learning for multi-agent cooperative control of connected autonomous vehicles. *Computer-Aided Civil and Infrastructure Engineering* 36, 7 (2021), 838–857.

[11] Jiqian Dong, Sikai Chen, Yujie Li, Runjia Du, Aaron Steinfeld, and Samuel Labi. 2021. Space-weighted information fusion using deep reinforcement learning: The context of tactical control of lane-changing autonomous vehicles and connectivity range assessment. *Transportation Research Part C: Emerging Technologies* 128 (2021), 103192.

[12] Runjia Du, Sikai Chen, Jiqian Dong, Paul Young Joun Ha, and Samuel Labi. 2021. GAQ-EBkSP: A DRL-based Urban Traffic Dynamic Rerouting Framework using Fog-Cloud Architecture. In *2021 IEEE International Smart Cities Conference (ISC2)*. IEEE, 1–7.

[13] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph neural networks for social recommendation. In *The world wide web conference*. 417–426.

[14] Jerome H Friedman. 2002. Stochastic gradient boosting. *Computational statistics & data analysis* 38, 4 (2002), 367–378.

[15] Fernando Gama, Antonio G Marques, Geert Leus, and Alejandro Ribeiro. 2018. Convolutional neural network architectures for signals supported on graphs. *IEEE Transactions on Signal Processing* 67, 4 (2018), 1034–1049.

[16] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *International conference on machine learning*. PMLR, 1263–1272.

[17] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 249–256.

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

[19] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction*. Vol. 2. Springer.

[20] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. 2013. *Applied logistic regression*. Vol. 398. John Wiley & Sons.

[21] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2013. An introduction to statistical learning (Vol. 112, p. 18).

[22] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[23] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[24] Yehuda Koren, Steffen Rendle, and Robert Bell. 2022. Advances in collaborative filtering. *Recommender systems handbook* (2022), 91–142.

[25] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 7, 1 (2003), 76–80.

[26] Jiangzhou Liu and Li Duan. 2021. A survey on knowledge graph-based recommender systems. In *2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, Vol. 5. IEEE, 2450–2453.

[27] Alexey Natekin and Alois Knoll. 2013. Gradient boosting machines, a tutorial. *Frontiers in neurorobotics* 7 (2013), 21.

[28] Grégoire Pacreau, Edmond Lezmi, and Jiali Xu. 2021. Graph Neural Networks for Asset Management. *Available at SSRN* (2021).

[29] Lily Qiu and Ivo Welch. 2004. Investor sentiment measures.

[30] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[31] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. 2007. Collaborative filtering recommender systems. In *The adaptive web*. Springer, 291–324.

[32] David I Shuman, Sunil K Narang, Pascal Frossard, Antonio Ortega, and Pierre Vandergheynst. 2013. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE signal processing magazine* 30, 3 (2013), 83–98.

[33] Carolin Strobl, James Malley, and Gerhard Tutz. 2009. An introduction to recursive partitioning: rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological methods* 14, 4 (2009), 323.

[34] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 417–426.

[35] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge graph convolutional networks for recommender systems. In *The world wide web conference*. 3307–3313.

[36] Shoujin Wang, Liang Hu, Yan Wang, Xiangnan He, Quan Z Sheng, Mehmet Orgun, Longbing Cao, Nan Wang, Francesco Ricci, and Philip S Yu. 2020. Graph learning approaches to recommender systems: A review. *arXiv preprint arXiv:2004.11718* (2020).

[37] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval.* 165–174.

[38] Ziteng Wu, Chengyun Song, Yunqing Chen, and Lingxuan Li. 2021. A review of recommendation system research based on bipartite graph. In *MATEC Web of Conferences*, Vol. 336. EDP Sciences, 05010.

[39] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 974–983.

[40] Sheng Zhang, Weihong Wang, James Ford, Fillia Makedon, and Justin Pearlman. 2005. Using singular value decomposition approximation for collaborative filtering. In *Seventh IEEE International Conference on E-Commerce Technology (CEC'05).* IEEE, 257–264.

[41] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Nguyen Hung, Zi Huang, and Lizhen Cui. 2020. Gcn-based user representation learning for unifying robust recommendation and fraudster detection. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval.* 689–698.