Composite recommendations with heterogeneous graphs

Naomi Rohrbaugh[1] and Edgar Ceh-Varela[2]

[1] University of Pittsburgh, Pittsburgh, PA, USA. ncr19@pitt.edu
[2] Eastern New Mexico University, Portales, NM, USA. eduardo.ceh@enmu.edu

Abstract. Recommender systems (RS) help users to deal with the problem of information overload. These systems suggest items to users based on their preferences. Traditionally, RS recommend a single item to the user, such as a movie, a song, or a place to eat. In some real-life situations, the user's preference for a particular item could be influenced by the entity (e.g., a brewery, a restaurant) that sells or produces the item (e.g., a beer, a hamburger). For example, when suggesting a hamburger, the recommendation might be enhanced by considering the user's liking for the restaurant that serves it. Therefore, the user's preference for the entity's characteristics must also be considered when recommending an item. This paper presents a recommender system model for composite recommendations. We define a *composite recommendation* as the recommendation of items with a "Has-a" relationship. For example, a restaurant "has-a" hamburger or a brewery "has-a" beer. Therefore, our model can recommend a tuple $\langle entity, item \rangle$ instead of a single item as traditional RS. As a basis, we use *metapath2vec* to obtain node embeddings from a heterogeneous graph. We formed the heterogeneous graph using features and user-entity, user-item, and entity-item interactions. The node embeddings for entities with a composite relationship are aggregated during recommendation to account for the user's preference for these entities. Our proposed model was tested with a real-life dataset composed of users, breweries, and beers. The results show that our proposed approach obtains better results than a baseline model, which does not consider the composite relationship.

Keywords: Recommender System · Graph Neural Network · Composite Recommendation.

1 Introduction

Recommender systems (RS) are valuable tools that use information related to a user, such as behavior, demographics, and social information, to predict a user's preferences for items [3]. Businesses with online platforms and social media websites often use RS to increase profitability by inferring users' preferences from past interactions [25]. Traditional RS present to customers a list of suggestions. Each suggestion consists of a single type of item, such as a book or a movie. These RS use the user-item interactions to make recommendations [16]. Other

types of RS are used to a provide personalized recommendation of a point of interest (POI) [8]. These POIs can be anything, including restaurants, hotels, parks, or museums. In this type of RS, the recommendation is made by analyzing the user's interactions with different POIs.

There has been a recent increase in the use of graph neural networks (GNN) as part of RS to make accurate predictions of a user's preferences [25]. GNN do this prediction using the observed user-item interactions from the graph. Heterogeneous graphs could be formed by multi-typed nodes and edges [26]. In these types of graphs, representation learning is used to obtain a lower-dimensional vector representation (i.e., embeddings) for each node while preserving the graph structure. A well-known method to get node embeddings in heterogeneous graphs is *metapath2vec* [6]. *Metapath2vec* uses random walks through the nodes and edges of a graph to learn vector representations of each node. The algorithm uses metapaths [6, 22] to capture the node relations. As similar node representations are placed close together in the latent space, the proximity of nodes can be used in different applications such as node classification, link prediction, and recommender systems.

In this paper, we address a new type of recommendation called composite recommendations. We use composite items to refer to items with a "Has-a" relationship (e.g., restaurant-hamburger, brewery-beer). For example, when recommending a hamburger, the recommendation could be improved by accounting for user preference of the restaurant that sells the food (i.e., the restaurant "Has-a" hamburger). On the one hand, a hamburger could be perfectly suited to a user's preferences but is sold at a restaurant with poor service or unhygienic practices, where the user does not want to eat. On the other hand, a restaurant might have the perfect vibe for a user but does not sell a hamburger that the user would enjoy. To remedy this predicament, the recommendation must consider the user's preferences for both restaurant and hamburger. Thus, composite recommendations that account for users' preferences for items in a "Has-a" relationship are needed.

We use the *metapath2vec* algorithm to obtain the node representations of a heterogeneous graph containing users, vendors, and items to produce composite recommendations. Then, our model performs a feature aggregation step to incorporate the user's representation and preferences for the venues that have a "Has-a" relationship with the items. The resulting user representation is then used to find a list with top-n items to be recommended.

The remainder of this paper is organized as follows: Section 2 presents the related literature. The problem is defined in Section 3. Section 4 details the proposed method. Section 5 outlines the experimental settings used to test our proposed model. The results showing the validity of our proposed method and its effectiveness are in Section 6. Finally, our conclusions are in Section 7.

2 Related Work

Recommender systems (RS) suggest new items or places to users based on past item interactions or using the interactions of similar users [4]. [1] proposes a meal recommendation system that the food chain sector may use to continuously update the options from their menu, based on customer likes, interests, and order history. In [5], user ratings and user reviews from an online beer community are used to recommend a beer to a user. Similarly, [11] presents a beer recommender system for Android mobile devices. This system uses fuzzy ontologies to represent pieces of information and semantic reasoners to infer implicit knowledge. In [12], a system to recommend POIs is presented. This system uses the relationship of textual information to model underlying patterns for POI recommendations. Similarly in [2], a POI recommender system is created by modeling users and POIs using the aspects posted on user reviews as a bipartite graph. In [7], a restaurant recommender system is proposed using user-based collaborative filtering. This system uses ratings given by users attending and restaurant location data. Likewise, in [10], an intelligent decision support model was created to recommend restaurants for individuals or groups. It uses features such as customer interest, price/budget, distance, and taste rating.

Recommendations can be generated using a bipartite graph where nodes are the users and items, and a link between these nodes represents an interaction between a user and an item. GNN has recently been used to learn user and item representations (i.e., embeddings) based on their features and interaction [25].

Representation learning assigns nodes in a graph to a lower-dimensional space while preserving the structure of the graph. Several methods have been used to obtain node representations. *Node2vec* [9], based on *DeepWalk* [18], defines a node's graph neighborhood and uses a biased random walk procedure to explore neighborhoods. This procedure allows the mapping of nodes to a low-dimensional space of features that maximizes the likelihood of preserving graph neighborhoods of nodes. Another approach based on *DeepWalk* is *"LINE"* [24]. This model learns the embedding of a graph while preserving the node's first- and second-order proximities (i.e., local and global graph structures).

However, there could be multiple types of nodes with different features and structural positions in a graph. The complication of making a GNN with heterogeneous nodes has inspired past researchers to develop graph embedding frameworks to preserve semantic relationships between different types of nodes [6].

A *heterogeneous graph* is defined in [6] as the graph $G = (V, E, T)$, where each node v has a mapping function $\phi(v) : V \rightarrow T_V$, and each edge e has a mapping function $\varphi(e) : E \rightarrow T_E$. The sets of objects and relation types are denoted as T_V and T_E, respectively, where $|T_V| + |T_E| > 2$.

Applying homogeneous graph methods for node representations on heterogeneous graph produce sub-optimal representations [25]. The *metapath2vec* [6] algorithm is a state-of-the-art method for representation learning of heterogeneous nodes. Metapath2vec uses latent space representations to place similar nodes close together without directly connecting them. This representation is

accomplished by specifying *metapaths* used on the random walks through the graph.

Some works have used the node representations obtained by metapath2vec to make recommendations. In [22], a recommender system is presented to recommend authors to any author in DBLP for citation purposes. The system creates a Heterogeneous User-Item (HUI) graph, and it is traversed using metapath2vec to capture the structural and semantic relationships between nodes. Similarly, [21] uses the node embeddings obtained from metapath2vec to recommend items and friends. The model uses cosine similarity between the node embeddings to generate a recommendation list.

The works presented in this section only recommend a single item (i.e., food, beer) and a single POI (i.e., restaurant). They do not consider how the user preference for the venue influences the user preference for a particular item being sold or produced in that place.

3 Problem Statement

We call composite items i and j to those items having a "Has-a" relationship.

$$i \xrightarrow{\text{"Has-a"}} j$$

For example, a brewery i *produces/sells* ("Has-a") a beer j. Moreover, i might have more than one item. In this case, we have $i \xrightarrow{\text{"Has-a"}} J_i$ (e.g., a restaurant sells burgers, burritos, and ice-creams). In a composite relationship, selecting an item j implicitly contains the item i that produces or contains j. Therefore, a tuple $\langle i, j \rangle$ can be generated containing the related items.

For example, when recommending a beer, the recommendation would be improved by accounting for the user preference of the brewery that produces that type of beer (i.e.,the brewery "Has-a" beer). On the one hand, a beer could be perfectly suited to a user's preferences but is sold/produced at a venue with poor service or lousy ambiance, aspects that are not to the user's liking. On the other hand, a brewery might have the perfect vibe for a user, but the kinds of beers produced in that venue do not meet user requirements. Therefore, the recommendation of a beer must consider the user's preferences for both breweries and beers.

Thus, having a set of users U, a set of items I, and a set of items J, where items from I have a composite relationship with items from J, the problem is to recommend an item j (e.g., beer) considering the characteristics of item i (e.g., brewery).

4 Proposed Approach

4.1 Generating a heterogeneous graph

Let us have the set of users U who have interacted with a set of items I (i.e., restaurants). Items I have a composite relationship (i.e., "Has-A") with the set of items J_i (i.e., item i has items J).

The elements of these sets and their relationship can be represented as nodes and edges in a heterogeneous graph $G = (V, E, T)$ (see Fig. 1 in Section 5.2). Each user node u_m and item node i_n are connected by an edge if user u_m has reviewed item i_n. Similarly, there is an edge between nodes u_m and j_l if there is a review from u_m to item j_l. Finally, an edge can be created between nodes i_n and j_l if i_n has an item j_l.

4.2 Obtaining the node embeddings

We are interested in the node embeddings for the elements in U, I, and J_i. To preserve the context of the nodes in an heterogeneous graph we employ a metapath strategy with unbiased random walks. With this strategy, we can incorporate semantic relationships between the different types of nodes.

The metapath2vec algorithm allows the construction of node representations as a vector considering the context of nodes conditioned on their types in a heterogeneous context.

A metapath is defined in a heterogeneous graph $G = (V, E, T)$ as [6, 22]

$$P : V_1 \xrightarrow{R_1} V_2 \xrightarrow{R_2} \cdots V_t \xrightarrow{R_t} V_{t+1} \cdots \xrightarrow{R_{l-1}} V_l \tag{1}$$

where $R = R_1 \circ R2 \circ \cdots \circ R_{l-1}$ defines the relationship between node types V_1 and V_l.

Moreover, the transition probability for each step i is calculated as follows

$$p(v^{i+1}|v_t^i, P) = \begin{cases} \frac{1}{|N_{t+1}(v_t^i)|} & (v^{i+1},\ v_t^i) \in E, \phi(v^{i+1}) = t+1 \\ 0 & (v^{i+1},\ v_t^i) \in E, \phi(v^{i+1}) \neq t+1 \\ 0 & (v^{i+1},\ v_t^i) \notin E \end{cases}$$

where $v_t^i \in V_t$ and $N_{t+1}(v_t^i)$ denotes the V_{t+1} type of node neighborhood of v_t^i. This indicates that the walker's direction is determined by the metapath defined in Eq. 1. As metapaths are usually symmetrical [22,23], the transition probability can be simplified using recursion as

$$p(v^{i+1}|v_t^i) = p(v^{i+1}|v_1^i), if t = 1$$

4.3 Feature aggregation

For clarity, we use bold lowercases for vectors and bold uppercases for matrices. For each user u_m who has not reviewed an item i_n, we compute a score based on cosine similarity (CS) and select the top-k items with the highest score. CS is suitable for metric learning given that the resulting similarity measure is always within the range of -1 and $+1$ [17]. A value close to -1 indicates low similarity, while a value close to $+1$ shows high similarity. Therefore, our score is defined as

$$score = CS(\mathbf{u_m}, \mathbf{i_n}) = \frac{\mathbf{u_m} \cdot \mathbf{i_n}}{||\mathbf{u_m}|| \cdot ||\mathbf{i_n}||} \tag{2}$$

where $\mathbf{u_m}$ and $\mathbf{i_n}$ are the user and item embedding vectors, respectively. After this step, we obtain a set $I^k_{u_m}$ with the k items (e.g., breweries) having the highest scores for user u_m.

To incorporate the features of the set of items $I^k_{u_m}$ having the set of items J_i (e.g., set of items J being sold by the business i) into a user's representation, we use the following:

$$\mathbf{u'_m} = \frac{1}{|I^k_{u_m}| + 1} \cdot \left(\mathbf{u_m} + \sum_{I^k_{u_m}} \mathbf{i_k} \right) \tag{3}$$

where $\mathbf{u'_m}$ is the new user representation (i.e., embedding) having the embeddings of the most preferred items i incorporated. $I^k_{u_m}$ is the set of the k most similar items i to u_m, $|\cdot|$ is the cardinality of the set, $\mathbf{u_m}$ is the user's embedding, and $\mathbf{i_k} \in \mathbf{I^k_{u_m}}$ the item's embedding.

4.4 Composite recommendations

Traditionally, a dot product has been used to combine embeddings and calculate similarity [19]. To capture the interest of a user u_m for an item j_l, we use a generalization of matrix factorization, which uses the dot product [14] for recommendations.

$$\mathbf{r_{u_m}} = \mathbf{u'^{\top}_m} \bullet \mathbf{J_{I^k_{u_m}}} \tag{4}$$

where $\mathbf{r_{u_m}}$ is a vector of scores for the items J_i not already rated by the user u_m, $\mathbf{J_{I^k_u}}$ is a matrix having the embeddings for each item j belonging to the set $I^k_{u_m}$, $\mathbf{u'_m}$ comes from Eq. 3, and $\bullet$ is the dot product.

Finally, $\mathbf{r_{u_m}}$ is sorted and the top-n items j are used in a tuple with their corresponding item i (i.e, $\langle i, j \rangle$), in a recommendation list.

5 Experimental settings

5.1 Dataset

To evaluate our proposed model, we generated a dataset with real-world data collected for two weeks in June 2021 from *BeerAdvocate.com*. The dataset contains information of users, breweries, and beers. For the study, we selected breweries from the state of California, USA. Moreover, the dataset includes user-beer and user-brewery interactions (i.e., reviews and ratings). Given that a brewery has beers, we can use this relationship for our composite recommendation.

Our dataset consists of 23,324 users, 959 breweries, and 39,349 beers. In total, we collected 273,831 records for user-beer interactions and 5,359 records for user-brewery interactions. On average, each brewery has 54.62 beers, each user has reviewed 12.07 beers and 2.73 breweries, and each beer has 17.20 user reviews. We split the dataset into 70% for training and 30% for testing.

5.2 Graph creation

We represent the dataset as a heterogeneous graph. We use the user IDs, beer IDs, and brewery IDs as node labels. Moreover, three different types of edges are created to represent the interaction between the different nodes (i.e., *User-Brewery*, *User-Beer*, *Beer-Brewery*). Table 1 shows the different features for the node types and for two of the edges (i.e., *user-brewery* and *user-beer*). The *beer-brewery* edge does not have additional features.

Table 1: Node and Edge Features

B(rewery)	*b(eer)*	*U(ser)*	*U-B*	*U-b*
Name	Name	Username	Username	Username
Beer Rating	Style	Location	Location	Location
# Beers Sold	ABV		Rating	Rating
Rating	Score		Vibe	Taste
Location	Location		Quality	Smell
	Availability		Service	Feel
	Rating		Selection	Look
			Food	Overall

Fig. 1 shows a representation of the heterogeneous graph. In total, the graph has 318,539 edges.

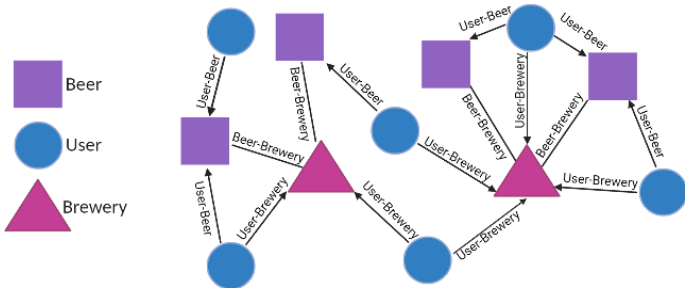

Fig. 1: Heterogeneous Graph

5.3 Node representations

To get each node representation (i.e., embeddings), we use the metapath2vec algorithm with symmetrical metapaths (i.e., same first and last node type). We use an embedding dimension of 64, a window's size of 5, 100 random walks per node, and each random walk with a length of 1. These values were selected after

testing different ranges. A set of random walks through the metapaths can train the GNN in a way that preserves structural correlations between nodes [6].

The following metapaths were considered (see Section 6.6): *Beer-Brewery-Beer*, *Beer-User-Beer*, *User-Brewery-User*, *User-Beer-User*, and *User-Beer-Brewery-Beer-User*.

Fig. 2 shows in a 2D space the embeddings for users (in green), breweries (in purple), and beers (in yellow). From this figure, we can observe clusters of the different nodes. These clusters indicate the similarity existing among nodes. For example, users with similar preferences given the beers and breweries rated will be placed in the same cluster of nodes.

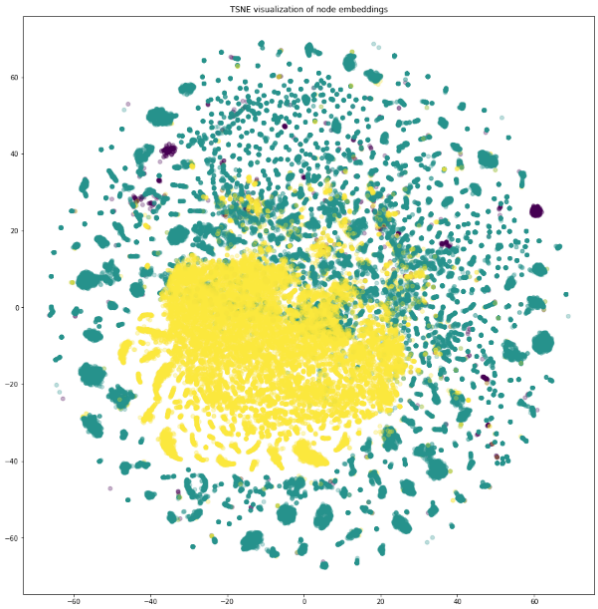

Fig. 2: Node Embeddings Visualization. Users in green, breweries in purple, and beers in yellow. (The figure is best understood in color.)

5.4 Evaluation metrics

We use *Hit Ratio* (*HR@n*) [13] to evaluate the performance of the composite recommendations. In this metric, n represents the length of the recommendation list. *HR@n* is computed by determining whether a hit exists within the top-n items in the list. The hit ratio is defined as [15]

$$HR@n = \frac{1}{U_T} \sum_{u=1}^{U_T} \mathbf{1}_{R_u \leq n} \tag{5}$$

where U_T is the set of test users, R_u indicates the relative rank of the item in the recommendation list, and $\mathbf{1}_X$ indicates the event X ($\mathbf{1}_X = 1$ iff X is a hit and 0 otherwise.)

Having an *HR@n* of 1 indicates that the recommender system can always recommend the items from the test set. On the other hand, a value of 0 indicates that the system is not able to find any of the test items [20].

5.5 Baseline

To compare our model, we use the model proposed in [21] as the baseline. This model also uses the metapath2vec algorithm to get the node embeddings. Then, using the user's embeddings, it recommends the top-n most similar items using cosine similarity.

6 Experimental Results

6.1 Model's performance

We tested the composite and baseline recommender systems with the test set. For our tests, we varied the size of the recommendation list (i.e., top-n) from 1 to 20 items. In our recommendation process, we use a value of $k = 7$ for Equation 3. Similarly, we ran the tests five times for each model, and the average is presented. Table 2 displays the average HR@n for our composite model and the baseline model when used for beer recommendations

Compared to the baseline model, our composite model consistently had significantly higher performance, regardless of the size of the recommendation list (i.e., top-n). These results indicate that integrating the information for the breweries that are more likely to be liked by the user to the user profile improves the recommendation for the beers that the user will like. The results align with our initial hypothesis that the user preference for a particular item could be influenced by the venue's characteristics and user preference for those characteristics.

Table 2: HR@n Performance comparison for beer recommendations varying the size of the recommendation list

	Baseline	Composite (ours)
n=1	0.0045	**0.0376**
n=3	0.0195	**0.1030**
n=5	0.0304	**0.1537**
n=7	0.0405	**0.1802**
n=10	0.0511	**0.1983**
n=15	0.0656	**0.2221**
n=20	0.0768	**0.2328**

6.2 Effect of the number of similar items (k)

We vary the value of k to determine how this value affects the model's performance. The value of k determines how many item i representations (e.g., breweries) are aggregated with the user representation.

Fig. 3 shows the results of HR@n when varying the value k for beer recommendations while fixing the size list to $n = 10$. We can see that the best result are obtained when k is between $k = 5$ to $k = 7$. Low values of k seem not to completely capture the characteristics of the most likely breweries, and higher values of k start introducing confusion to the model.

6.3 Effect of the embedding dimension

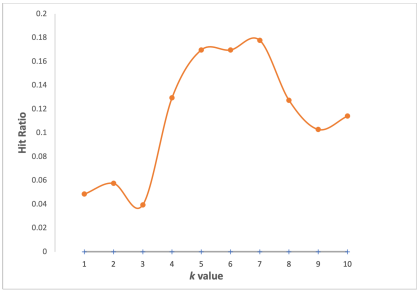

Fig. 3: Effect of the value k

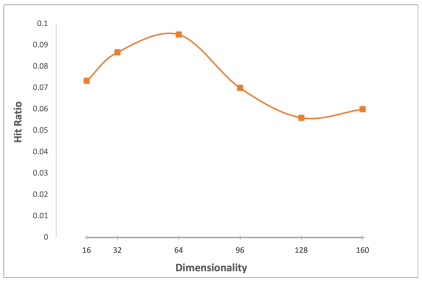

Fig. 4: Effect of Embedding Dimension

Fig. 4 shows the model's performance when varying the dimension for the embeddings.

The results show that the model performs better with an embedding dimension of 64. These results indicate that embeddings of lower dimensions cannot capture the full extent of the graph with latent space representations. Moreover, it also shows that a higher embedding dimension is likely to introduce noise to the node representations.

6.4 Effect of the random walk length

Fig. 5 shows the model performance when varying the length of the random walk for the metapath2vec algorithm. The results show that with random walk lengths of 100, the model performs the best. From the results, we also can see that the performance is reduced as the length of the walk grows. This reduction in performance could result from uncertainty introduced to the model because of the use of information of less similar nodes.

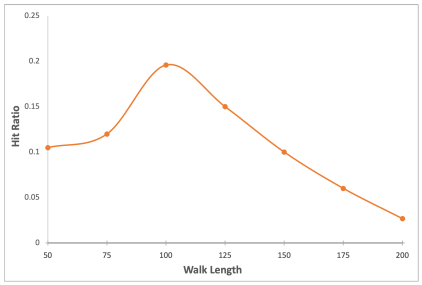

Fig. 5: Effect of Walk Length

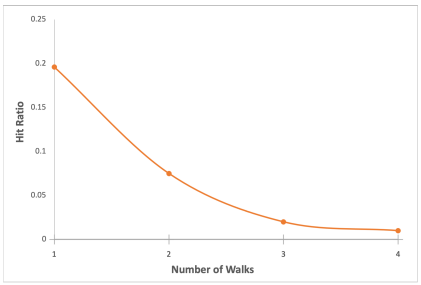

Fig. 6: Effect of Number of Walks

6.5 Effect of the number of random walks

Fig. 6 shows the model's performance when varying the number of random walks for the metapath2vec algorithm. From the results, we can see that only one walk is enough. More walks per node could have introduced too much inconsistency to the system, causing decreased performance.

6.6 Effect of the metapaths

We wanted to test if the selection of metapaths influences the model's performance. We initially defined the following set of metapaths: *Beer-Brewery-Beer*, *Brewery-User-Brewery*, *Beer-User-Beer*, *User-Brewery-User*, *User-Beer-User*, *User-Beer-Brewery-Beer-User*, and *User-Beer-Brewery-User*. To choose the best subset of metapaths from the set initially defined, we excluded one metapath at a time and tested the model's performance using the remaining metapaths. The resulting HR@n was compared to the average HR@n for all metapaths to determine whether excluding the metapath had improved the performance.

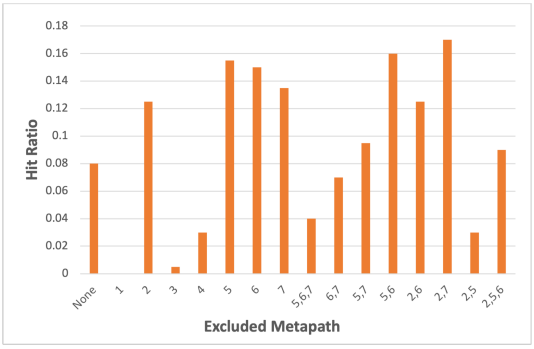

Fig. 7: Effect of Metapaths

Fig. 7 shows the model's performance when excluding metapaths from the initial set. The results show that the model's performance increases when removing the individual metapaths 2, 5, 6, and 7.

We further test if excluding combinations of these metapaths affected the model's performance. The results from these tests show that excluding the combination of metapaths 2 and 7 yielded the best results. As a result, the following metapaths were used: *Beer-Brewery-Beer, Beer-User-Beer, User-Brewery-User, User-Beer-User, User-Beer-Brewery-Beer-User*.

Using a set of diverse metapaths seemed to maximize results. This result was made clear when removing similar metapaths caused an increase in performance.

Surprisingly, not using metapaths with breweries as endpoints improves the performance of the recommendations. This outcome could happen because the brewery's representation is implicitly obtained by the preferences of the users who attend and the characteristics of the beers sold, mainly because each beer belongs only to one brewery.

7 Conclusions

This work proposes a composite recommender system model that accounts for user preferences of items in a "Has-a" relationship (e.g., brewery has a beer). Therefore, our model can recommend a tuple $\langle entity, item \rangle$ instead of a single item as traditional RS.

The proposed system uses metapaths based on random walks to improve the Graph Neural Network's ability to model the dataset. The recommender system model accurately captures user preferences by embedding a heterogeneous graph with nodes for users, vendors, and items. The use of feature aggregation to account for a user's vendor preferences in selecting an item improves the recommendation by considering all aspects of the recommended set. The composite recommender system significantly outperformed the baseline model.

Acknowledgment

This work is partially supported by the NSF grant #1950121.

References

1. Ahuja, K., Goel, M., Sikka, S., Makkar, P.: What-to-taste: A food recommendation system (2020)
2. Baral, R., Zhu, X., Iyengar, S., Li, T.: Reel: Review aware explanation of location recommendation. In: Proceedings of the 26th conference on user modeling, adaptation and personalization. pp. 23–32 (2018)
3. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A.: Recommender systems survey. Knowledge-based systems **46**, 109–132 (2013)

4. Ceh-Varela, E., Cao, H.: Recommending packages of multi-criteria items to groups. In: 2019 IEEE International Conference on Web Services (ICWS). pp. 273–282. IEEE (2019)
5. Chinchanachokchai, S., Thontirawong, P., Chinchanachokchai, P.: A tale of two recommender systems: The moderating role of consumer expertise on artificial intelligence based product recommendations. Journal of Retailing and Consumer Services **61**, 102528 (2021)
6. Dong, Y., Chawla, N.V., Swami, A.: metapath2vec: Scalable representation learning for heterogeneous networks. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. pp. 135–144 (2017)
7. Fakhri, A.A., Baizal, Z., Setiawan, E.B.: Restaurant recommender system using user-based collaborative filtering approach: A case study at bandung raya region. In: Journal of Physics: Conference Series. vol. 1192, p. 012023. IOP Publishing (2019)
8. Gottapu, R.D., Monangi, L.V.S.: Point-of-interest recommender system for social groups. Procedia computer science **114**, 159–164 (2017)
9. Grover, A., Leskovec, J.: node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 855–864 (2016)
10. Hartanto, M., Utama, D.N.: Intelligent decision support model for recommending restaurant. Cogent Engineering **7**(1), 1763888 (2020)
11. Huitzil, I., Alegre, F., Bobillo, F.: Gimmehop: A recommender system for mobile devices using ontology reasoners and fuzzy logic. Fuzzy Sets and Systems **401**, 55–77 (2020)
12. Khanthaapha, P., Pipanmaekaporn, L., Kamonsantiroj, S.: Topic-based user profile model for poi recommendations. In: Proceedings of the 2nd International Conference on Intelligent Systems, Metaheuristics & Swarm Intelligence. pp. 143–147 (2018)
13. Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 426–434 (2008)
14. Koren, Y., Bell, R., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
15. Li, D., Jin, R., Gao, J., Liu, Z.: On sampling top-k recommendation evaluation. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 2114–2124 (2020)
16. Mu, R.: A survey of recommender systems based on deep learning. Ieee Access **6**, 69009–69022 (2018)
17. Nguyen, H.V., Bai, L.: Cosine similarity metric learning for face verification. In: Asian conference on computer vision. pp. 709–720. Springer (2010)
18. Perozzi, B., Al-Rfou, R., Skiena, S.: Deepwalk: Online learning of social representations. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 701–710 (2014)
19. Rendle, S., Krichene, W., Zhang, L., Anderson, J.: Neural collaborative filtering vs. matrix factorization revisited. In: Fourteenth ACM Conference on Recommender Systems. pp. 240–248 (2020)
20. Sieg, A., Mobasher, B., Burke, R.: Improving the effectiveness of collaborative recommendation with ontology-based user profiles. In: proceedings of the 1st International Workshop on Information Heterogeneity and Fusion in Recommender Systems. pp. 39–46 (2010)

21. Sowmya, A., Shebin, K.M., Mohan, A., et al.: Social recommendation system using network embedding and temporal information. In: 2020 5th International Conference on Computing, Communication and Security (ICCCS). pp. 1–7. IEEE (2020)
22. Sun, Y., Han, J.: Mining heterogeneous information networks: principles and methodologies. Synthesis Lectures on Data Mining and Knowledge Discovery **3**(2), 1–159 (2012)
23. Sun, Y., Han, J., Yan, X., Yu, P.S., Wu, T.: Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. Proceedings of the VLDB Endowment **4**(11), 992–1003 (2011)
24. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., Mei, Q.: Line: Large-scale information network embedding. In: Proceedings of the 24th international conference on world wide web. pp. 1067–1077 (2015)
25. Wu, S., Sun, F., Zhang, W., Cui, B.: Graph neural networks in recommender systems: a survey. arXiv preprint arXiv:2011.02260 (2020)
26. Zhang, C., Song, D., Huang, C., Swami, A., Chawla, N.V.: Heterogeneous graph neural network. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 793–803 (2019)