Health-guided recipe recommendation over knowledge graphs

Diya Li a, Mohammed J. Zaki a,*, Ching-hua Chen b

a Computer Science Department, Rensselaer Polytechnic Institute, Troy, NY, USA
b Center for Computational Health, IBM Research, Yorktown Heights, NY, USA

Abstract

While the availability of large-scale online recipe collections presents opportunities for health consumers to access a wide variety of recipes, it can be challenging for them to discover relevant recipes. Whereas most recommender systems are designed to offer selections consistent with users' past behavior, it remains an open problem to offer selections that can help users' transition from one type of behavior to another, intentionally. In this paper, we introduce health-guided recipe recommendation as a way to incrementally shift users towards healthier recipe options while respecting the preferences reflected in their past choices. Introducing a knowledge graph (KG) into recommender systems as side information has attracted great interest, but its use in recipe recommendation has not been studied.

To fill this gap, we consider the task of recipe recommendation over knowledge graphs. In particular, we jointly learn recipe representations via graph neural networks over two graphs extracted from a large-scale Food KG, which capture different semantic relationships, namely, user preferences and recipe healthiness, respectively. To integrate the nutritional aspects into recipe representations and the recommendation task, instead of simple fusion, we utilize a knowledge transfer scheme to enable the transfer of useful semantic information across the preferences and healthiness aspects. Experimental results on two large real-world recipe datasets showcase our model’s ability to recommend tasty as well as healthy recipes to users.

1. Introduction

Food is critical to human beings as it is vital for good health and ultimately for life itself. The recording and sharing of recipes in online repositories has led to fast growth of food computing. Food recommendation aims to provide a list of ranked food items for users to meet their personalized needs. In recent years, introducing a knowledge graph (KG) into the recommender system as side information has attracted great interest since a KG can provide extra information and can be used to suggest potential relations among items [1–4]. We posit that recipe items and their attributes (i.e., ingredients and categories) mapped into the KG can enable a food recommender system to better understand the latent mutual relations between them and make the recommendation results more explainable. For example, a user who loves Tom Yum Soup may have interest in Tom Kha Soup since the distance of the two items is quite close in a food KG, given that they have similar ingredients and both are Thai style, whereas the suggestion is hard to extract from the user’s preference list if only one of the recipes is logged. However, KG-based recipe recommendation has not been widely studied. To fill this gap, we utilize a public user–item interaction dataset [5] scraped from (comprising 180K+ recipes and 700K+ recipe reviews) along with the FoodKG knowledge graph [6] (comprising around 67 million triples spanning a food ontology and recipes with multiple attributes, including nutrition). In addition, we create a new recipe recommendation dataset generated from user records in MyFitnessPal [7]. In our KG-based recommendation, the recipe representations are enriched by aggregating embeddings of multi-hop neighbors in the recipe KG to predict the user’s preference via the refined user and candidate recipe representations. To guide the information flow in the KG, we adopt a graph neural network (GNN) based approach to adjust the weights between different neighbors during the propagation process.

Compared to other recommender systems applied in general domains such as movies, music, or products, food recommendation is highly relevant to human health and thus plays a critical role in human dietary choice [8]. Food recommendation involves more complex and multi-faceted information. In addition to matching user preferences, it is therefore equally important to include healthiness scores. Due to unhealthy eating habits, such as increased intake of food containing high energy or fat, a growing proportion of the global population is becoming overweight or obese [9]. Without considering the health aspect, a recommender system will keep suggesting calorie-rich foods for someone with

* Corresponding author.
E-mail addresses: 916lidiya@gmail.com (D. Li), zaki@cs.rpi.edu (M.J. Zaki), chinghua@us.ibm.com (C. Chen).

https://doi.org/10.1016/j.websem.2022.100743
1570-8268/© 2022 Elsevier B.V. All rights reserved.
unhealthy eating habits. Studies in nutrition science have shown that proper nutrition and health labels help people to make better food choices [10]. An ideal food recommendation system should offer a trade-off between user preferences and nutrition requirements, recommending both “healthy” and “tasty” foods. Existing health-guided food recommendation methods balance user’s food preference and the health goal via simple fusion [11,12] or by adding nutrition constraints referring to particular health guidelines [13,14]. In contrast, while extracting a user’s interest from real-world user–item interaction datasets (here item refers to a recipe), we also consider the health aspects using an item’s nutrition information. To make the health aspect more adaptable in recommendation, we learn two different item representations through two special graphs extracted from a KG. In these two graphs, items/recipes have different semantics with regards to similarity and healthiness. In order to integrate the health aspect into the final recipe representation and the following recommendation, we utilize a knowledge transfer scheme, that enables useful semantic information to be exchanged across the two representations. Therefore, the nutrition information is incorporated into the food recommender system for constraint optimization and computing. To fairly evaluate the recommendation results, we design new criteria to consider both the healthiness and preferences. Our main contributions are:

- We highlight the importance of recommending food considering the health aspects and incorporating knowledge graphs as side information to enrich the item representation. To the best of our knowledge, this is the first work to do health-based recipe recommendation over KGs.
- We separately model user preference and food healthiness in two differently structured graphs obtained from a large-scale Food KG where items have varied semantics representing similarities and healthiness. Furthermore, a knowledge transfer mechanism is adopted to let the two aspects share useful information and thus to benefit each other. The final item representations are enriched by the knowledge from the health aspects.
- We examine our model on two real-world recipe datasets. We also design a new criterion to evaluate the healthiness of the recommended results. Our experiments demonstrate the effectiveness of our model and its interpretability in recommending both “healthy” and “tasty” recipes.

2. Related work

2.1. Health-guided food recommendation

Because of the overload of information with the rapid development of the internet, it has become hard for users to pick out what interests them among a lot of choices. Recommender systems have been applied in many scenarios to improve the user experience. Among the multiple applications of recommender systems, food recommendation is special since the health aspect serves a crucial role. Previous methods have tried to incorporate healthiness into the recommendation process by substituting healthier ingredients [15,16], adding calorie counts as a manually adjusted feature [12], and incorporating nutritional facts directly as linear constraints [11,13]. Trattner and Elsweiler [17] proposed a straightforward post-filtering approach which re-weights the scores of recommended recipes for a particular user based on recipe healthiness score. However, these methods balance a user’s food preference and the health aspects via simple combination operations, such as summation [12] and multiplication [17]. It is hard to optimize the trade-off between the two aspects. We argue that there is a need for more effective fusion methods. Our model jointly trains two representations with regards to user preference and food healthiness, and then adopts a knowledge transfer scheme to integrate the two aspects.

There are other studies targeting personalized recipe recommendation with specific health goals: Yang et al. [18] proposed a nutrient-based recipe recommender system to meet individuals’ nutritional expectations and specific dietary restrictions referring to user profiles and visual food image features. Chen et al. [14] modeled food recommendation as a constrained question answering task over a food knowledge graph. They did personalized recommendation by adding users’ dietary preferences and health guidelines as additional constraints. Our method has a different task setting, we recommend recipes directly over the KGs utilizing user–item (i.e., user–recipe) interactions.

2.2. KG-based recommender systems

Generating recommendations from a KG has attracted interest recently since KGs can improve the recommendation results as well as provide interpretability. Compared to traditional recommendation methods like collaborative filtering and content-based filtering, KG-based recommender systems can further alleviate the data sparsity and cold-start problems by incorporating external knowledge. Guo et al. [4] grouped KG-based recommendation methods into three categories: embedding-based methods that emphasize how KG embeddings are learned, connection-based methods that focus on the connection patterns in the KG, and propagation-based methods that address how item representations are refined in the propagation process. In particular, connection- and propagation-based methods can provide interpretability by examining the semantic and structural information in a KG.

Our work falls within the propagation-based paradigm, where the aim is to capture high-order relations between items in the KG by aggregating representations of their multi-hop neighbors [1,2,19–21]. Specifically, a graph is often extracted from the original KG by mapping the items in a users–items interaction dataset to their associated entities in the KG and selecting their multi-hop neighbors as related entities. The extracted graph then serves as an input for KG-based recommendation, where the aggregation function is usually implemented using various graph neural networks. For example, Wang et al. [19] proposed a measure to make the weight of each neighbor be user-specific by considering both the user representation and the item relation. As a follow-up approach, Wang et al. [3] added label smoothness regularization to solve the overfitting problem in [19]. Among recent works, Ma et al. [22] proposes a new model for recommendation in hyperbolic space that facilitates learning of hierarchical structure in KGs. Chen et al. [23] proposes a non-sampling based approach to KG learning, and Mu et al. [24] does disentangled learning (i.e., multi-aspect representations of users and items) of latent factors in KGs. Unlike these models that unify all the item attributes in a single graph extracted from the KG, we employ two different graphs constructed from the KG that refer to different semantics: the user preferences and the recipe/item healthiness. We then jointly learn these aspects via a unified model using graph neural networks. We show that our model that fuses both user preference and food healthiness outperforms state-of-the-art KG-based recommender systems.

3. Background

3.1. Food knowledge graph

A knowledge graph is a directed graph comprising subject–property–object triples (edges) that specify different types of
relationships (properties or predicates) between the nodes (subjects and objects). The KG can be represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$ is the set of $n$ (heterogeneous) entities (e.g., recipes, ingredients, meals, etc.). Entities are associated with attributes $\mathcal{X} = \{x_1, x_2, \ldots, x_L\}$, where $x_i$ represents the attributes of entity $v_i$. For example, recipes contain attributes like recipe category, nutrition, cook time, serving size, and so on. The edges $\mathcal{E}$ represent relationships between entities; each edge of the form $(v_h, r, v_l) \in \mathcal{E}$ indicates a relationship $r$, from head entity $v_h$ to tail entity $v_l$. For instance, (banana bread, consist_of, all purpose flour) states the fact that banana bread consists of all purpose flour.

Entities in $\mathcal{G}$ can be connected with a multi-hop relation path:

$v_0 \xrightarrow{r_1} v_1 \xrightarrow{r_2} \cdots \xrightarrow{r_l} v_l$

where $v_l \in \mathcal{V}$ and $\{v_i, r_{i+1}, v_{i+1}\} \in \mathcal{E}$. In this case, $v_l$ is the $l$-hop neighbor of $v_0$, which is denoted as $v_l \in N^l(v_0)$.

3.2. Food recommendation task

A typical recommendation task is to suggest a list of ranked unobserved items based on a user’s preference to meet the user’s needs. The final recommendation is generated by sorting the preference scores of items (i.e., recipes). We have a set of $M$ users $\mathcal{U} = \{u_1, u_2, \ldots, u_M\}$, a set of $N$ items $\mathcal{V}' = \{v_1, v_2, \ldots, v_N\}$, and the user–item interaction matrix $\mathcal{Y} \in \mathbb{R}^{M \times N}$. The user–item interaction matrix is defined according to users’ implicit feedback, where $y_{ui} = 1$ indicates that user $u_i$ has interest in item $v_l$; otherwise $y_{ui} = 0$. Given the user–item interaction matrix $\mathcal{Y}$ as well as the corresponding knowledge graph $\mathcal{G}$ where items in $\mathcal{V}'$ are mapped to nodes in $\mathcal{G}$ (so that $\mathcal{V}' \subseteq \mathcal{V}$), the recommender system first learns the representations of the user, denoted $u_i \in \mathbb{R}^d$, and candidate item, denoted $v_l \in \mathbb{R}^d$, based on $\mathcal{G}$. Then, a prediction function

$\hat{y}_{u,v} = \mathcal{F}(u, v | \mathcal{Y}, \mathcal{G})$

is learnt to model the preference of $u$ for $v$. Health-guided recommendation has to further model and incorporate nutritional information into the final recommendations. We list the key notations used in this paper in Table 1.

4. Health-guided food recommendation

The overall schematic of our health-guided recipe recommendation framework is shown in Fig. 1. Two special graphs – preference graph and health graph – are generated from the large-scale FoodKG knowledge graph [6], with respect to user preference and healthiness. Then two types of item representations are learned via two different graph neural networks. Finally, we adopt a knowledge transfer mechanism to share useful information between the preference and health representations for the items. Details are given next.

4.1. Preference and health graphs

For KG-based recommendation, the first step is to extract a relevant graph from a large KG. Recipes appearing in the user–item interaction dataset are first mapped to an external KG to find their associated entities. For recommendation in general domains like suggesting movies, books, or products, the relations between entities are usually predefined in their associated KGs, which allows one to extract multi-hop neighbors of related entities from the KG to create a graph for the recommendation method. However, in the food KG there are no direct edges between recipes. Therefore, for each item $v_l \in \mathcal{V}'$ in the user–item interaction matrix, we perform random walks with restart to find the $k$ most related recipe entities in the KG $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The $k$ most related entities can be denoted as $v_l = \{v_{l1}, v_{l2}, \ldots, v_{lk}\}$. Thus, the final set of relevant vertices in the KG is the set of all such items from the user-interaction items (recipes) and their related entities, i.e., $\mathcal{V} = \bigcup_{v_l} \mathcal{V}_l$ (where $v_l \in \mathcal{V}'$).

Preference graph: To perform a random walk among recipes in the food KG, we first define a potential edge between recipes $v_h$ and $v_l$ if the number of undirected paths between them is larger than $\theta_p$, with the length of undirected paths restricted to 3 hops (in practice, we set $\theta_p = 3$ and $l \leq 3$). The length constraint, $l \leq 3$ allows three conditions for determining an undirected path $p$: (i) recipe $v_h$ and $v_l$ have one common ingredient; (ii) ingredient $h$ in recipe $v_h$ is relevant to ingredient $i$ in $v_l$; (iii) recipe $v_h$ and $v_l$ are recommended for the same meal type.

For condition (i), we have a 2-hop relation path between $v_h$ and $v_l$ as:

$v_h \xrightarrow{r} i \xleftarrow{r} v_l$

where $r$ is the consist_of relation and $i$ is a common ingredient between recipes $v_h$ and $v_l$.\(^1\) For condition (ii), we get a 3-hop relational path between $v_h$ and $v_l$ as:

$v_h \xrightarrow{r} h \xrightarrow{r} i \xleftarrow{r} v_l$

where relation $r'$ is the substitutes_for relation in the food KG. For condition (iii), the 2-hop relation path is similar to condition (i), except that $r$ is the is_recommended_for_meal relation and $i$ is their common meal type. For instance, the recipe cream cheese apple pie and apple pie coffee cake have one common ingredient apple and they are both suggested to serve as snack. In addition, the cream cheese in cream cheese apple pie can be substituted with sour cream, which is contained in apple pie coffee cake with reference to the food KG. Thus, we put a potential edge between the two recipes.

After constructing direct potential edge among recipes, to perform a random walk over the recipes, the next step is to consider the edge weight with regards to recipe semantics in the KG. As a structured data type, the recipes consist of a list of ingredients, titles, cooking instructions, nutrition content, and category information. The edge weight for a random walk between recipes $v_h$ and $v_l$ is calculated as the overlap ratio in the recipe title and ingredients, since the two components can adequately represent the recipe itself. The recipe title and ingredients are collected by querying triples in KG. Specifically, the transition probability $P_(\theta_p)(v_h, v_l)$ between item $v_h$ and $v_l$ can be formulated as:

$P_(\theta_p)(v_h, v_l) = \lambda \cos(v_h^{itle}, v_l^{itle}) + (1 - \lambda) \cos(v_h^{ingre}, v_l^{ingre})$

(1)

where $v_h^{itle}$ is the tf-idf vector of the title, and $v_l^{ingre}$ is a binary vector of ingredients for an item ($a \in \{h, t\}$); and $\lambda \in [0, 1]$ is

\(^1\) We discard high frequency seasoning ingredients like salt, sugar, and oil to eliminate redundant edges.
Health graph: Based on the chosen related entities, \( v \) are preprocessed by lemmatization. Finally, related recipes are extracted from food KG and form a graph for the recommendation system. The construction of the graph filters out less related entities in the graph, facilitating mining high-order item relations for recommendation. The graph is built with regards to recipe similarities; we term it as the preference graph because general food recommendation is based on user preferences, which mainly tends to recommend similar foods. The preference graph is denoted as \( G^{(p)} = (V, E^{(p)}) \).

Health graph: Based on the chosen related entities, \( V = \{v_1, v_2, \ldots, v_l\} \), we construct the health-based graph for health-guided recommendations. We conduct a second random walk on the chosen entities, where the edge weight between two recipes is based on the title similarity and their nutrition content. Each recipe contains nutritional information in terms of macronutrients (e.g., fat, saturated fat, sugar, salt). The nutrition information is obtained from the food KG by inquiring the has_nutrition relation. To incorporate the numeric value of the macronutrients into representations, a nutritional quality rating is used, based on the international standards introduced in 2007 by the Food Standards Agency (FSA) (food.gov.uk). For each recipe, an FSA color-coded rating (red-bad, amber-caution or green-good) is computed for each of the macronutrients, proving a clear and understandable indication of how healthful the recipe is. Following previous work [25], we then map the FSA color ratings to numeric values (red-bad: 1, amber-caution: 2, green-good: 3) on a discrete scale, as healthiness scores. We let \( A_i = [a_1, \ldots, a_i] \) be \( N \) category values denoting the healthiness score of item \( v_i \), where \( N \) is the number of nutrients. Based on the motivation that, in addition to similarity, a recipe should have a higher edge weight towards a healthier recipe, the edge weight from recipe \( v_{\text{hr}} \) to \( v_i \) is regulated by their healthiness scores. The edge weight from recipe \( v_{\text{hr}} \) to \( v_i \) is increased if \( v_i \) is healthier than \( v_{\text{hr}} \), and vice versa. The health-based edge transition probability \( P_{\text{hr}}(v_i, v_j) \) between item \( v_i \) and \( v_j \) can be calculated as:

\[
\Gamma = \cos(v_i^{\text{title}}, v_j^{\text{title}})
\]

\[
P_{\text{hr}}(v_i, v_j) = \max \left( 0, \delta \cdot \Gamma + (1 - \delta) \frac{\sum A_{v_i} - \sum A_{v_j}}{2N} \right)
\]

where \( \delta \in [0, 1] \) is a scalar parameter, and \( \sum A_{v_i} \) is the summation of healthiness scores of an item \( (a \in \{h, t\}) \). Note that the maximum difference in health score over \( N \) nutrients is \( 2N \) (for good: 3 - bad: 1). Thus, we build a new graph where healthy recipes are clustered and have more connections in this graph. We term this graph as the health graph and denote it as \( G^{(h)} = (V, E^{(h)}) \). Note that the preference graph, \( G^{(p)} \), and the health graph, \( G^{(h)} \), share the same entity set while having different edges representing different semantics. As illustrated in Fig. 1, \( v_1 \) and \( v_2 \) are one hop and two hops away from item \( v_i \) in the preference graph, respectively. However, \( v_2 \) is pulled closer to \( v_i \) in the health KG as it is healthier than \( v_1 \).

4.2 Refinement of item representations

After obtaining the preference graph and the health graph, every entity in the two graphs can be regarded as a candidate item \( v_i \) for recommendation. The next step is to learn a higher order representation of the candidate item, denoted as \( v_i \in \mathbb{R}^d \), by aggregating embeddings of item \( v_i \)'s multi-hop neighbors \( N_i^l \) \( (l = 1, 2, \ldots, L) \) in both graphs. Here \( d \) represents the dimension of the latent representation. The graph convolution network (GCN) proposed by Kipf and Welling [26] is a widely adopted choice as the kernel for propagation [3, 19, 27]. With the usage of GCN, during the aggregation process, non-parametric weights are assigned to different neighbors to make the propagation relation-specific. Such mechanisms have been widely used in KG-based recommendation due to the motivation that a user will have distinct preferences for different relations.

4.2.1 Learning in the preference graph

We formulate the representation of item \( v_i \) from its \( l \)-hop neighbors in the preference graph as:

\[
\psi_{N_i^{l-1}}^{-1} = \gamma\left(\{\psi_{N_i^{l-1}}, j \in N_i\}\right)
\]

\[
\psi_i^{l-1} = g(\psi_i^{l-1}, \psi_{N_i^{l-1}}^{-1})
\]

where \( N_i \) denotes the neighbors of item \( v_i \), \( \psi_{N_i^{l-1}}^{-1} \) denotes the aggregated feature vector of \( N_i \), \( \gamma(\cdot) \) is the GCN aggregation function (over a node’s neighbors), \( g(\cdot) \) is the final aggregation function. There are many ways to aggregate \( \psi_i^{l-1} \) and \( \psi_{N_i^{l-1}}^{-1} \), such as summation [26], concatenation [28], or a hybrid operation [2]. Here we implement \( g(\cdot) \) in three different ways:

\[
\psi_i = \sigma(W (\psi_i^{l-1} + \psi_{N_i}^{-1}))
\]

Fig. 1. The framework of our proposed model. Two graphs (preference graph and health graph) are generated from the FoodKG. Then two types of item representations are learnt through GCN and GAT. Finally, a knowledge transfer mechanism is adopted to enable the transfer of useful semantic information across the preferences and healthiness aspects.
\[
\mathbf{v}'_i = \sigma \left( \sum_{j \in N_i} \alpha_{ij} \mathbf{W} \mathbf{v}^{-1}_j \right)
\]
(7)

\[
\mathbf{v}'_i = \sigma \left( \mathbf{W}_1 (\mathbf{v}^{-1}_i + \mathbf{v}^{-1}_{N_i}) + \sigma \left( \mathbf{W}_2 (\mathbf{v}^{-1}_i \odot \mathbf{v}^{-1}_{N_i}) \right) \right)
\]
(8)

where \(\mathbf{W}, \mathbf{W}_1\) and \(\mathbf{W}_2\) represent trainable weight matrices and \(\sigma(\cdot)\) is a ReLU activation function. The additional term in Eq. (8) compared to Eq. (6) makes the information being propagated sensitive to the affinity between \(\mathbf{v}^{-1}_i\) and \(\mathbf{v}^{-1}_{N_i}\). Typically, for GCN, it explicitly assigns a non-parametric weight \(w_j = 1/\sqrt{\text{deg}(v_i) \cdot \text{deg}(v_j)}\) to the neighbor \(v_j\) of \(v_i\) during the aggregation process \(\gamma(\cdot)\), where \(\text{deg}(v_i)\) denotes the degree of item \(v_i\). For recommendation, to make the weight of each neighbor \(j \in N_i\) user-specific, we note the trainable user representation as \(\mathbf{u} \in \mathbb{R}^d\) and define \(\gamma(\cdot)\) as:

\[
w_j = \frac{\exp(w_j)}{\sum_{k \in N_i} \exp(w_k)}
\]
(9)

\[
\mathbf{v}^{-1}_{N_i} = \gamma \left( (\mathbf{v}^{-1}_i, j \in N_i) \right) = \sum_{j \in N_i} \tilde{w}_j \mathbf{v}^{-1}_j
\]
(10)

\[
\mathbf{v}'_i = \sigma \left( \sum_{j \in N_i \cup \{i\}} \alpha_{ij} \mathbf{W} \mathbf{v}^{-1}_j \right)
\]
(12)

where \(\sigma(\cdot)\) is a non-linear activation function (e.g., ReLU), and \(\mathbf{v}^0 = \mathbf{v}_i\) is the initial representation of item \(v_i\). The attention weight \(\alpha_{ij}\) measures the connection strength between the item \(v_i\) and its neighbor \(v_j\) and it is calculated as:

\[
\alpha_{ij} = \text{softmax} \left( \phi \left( \mathbf{a}^T [\mathbf{W} \mathbf{v}^{-1}_i \parallel \mathbf{W} \mathbf{v}^{-1}_j] \right) \right)
\]
(13)

where \(\phi(\cdot)\) is a LeakyReLU activation function, and both \(\mathbf{a}\) and \(\mathbf{W}\) are learnable parameters. The softmax function ensures that the attention weights sum up to one over all neighbors of item \(v_i\). We further use multi-head attention with \(K\) heads to increase the model’s expressive capacity, obtaining \(\mathbf{v}'_i\) as a concatenation of the \(K\) heads:

\[
\mathbf{v}'_i = \left[ \sigma \left( \sum_{j \in N_i \cup \{i\}} \alpha_{ij}^{(k)} \mathbf{W}^{(k)} \mathbf{v}^{-1}_j \right) \right]_{k=1}^K
\]
(14)

where \((k)\) denotes the \(k\)th head. Under the umbrella of convolutional graph neural networks, a GAT model usually stacks multiple convolutional layers over the shallow feature embeddings (i.e., \(\mathbf{v}'_i\)), which extract high level information of both the item features and graph structure into the final-layer item embedding (i.e., \(\mathbf{v}'_i\)).

Note that the two GNNs (GCN and GAT) take different graph structures (\(G^{(p)}\) and \(G^{(h)}\)) and item attributes \(x\) as inputs. Therefore, for the same recipe item \(v_i\) in \(G^{(p)}\) and \(G^{(h)}\), we get the final preference-aspect representation \(\mathbf{v}^{(p)}_i \in \mathbb{R}^d\) via GCN and the health-aspect representation \(\mathbf{v}^{(h)}_i \in \mathbb{R}^d\) via GAT, capturing different semantics, and obtained from the last layer of GCN and GAT, respectively. Our method encodes different item semantics in different latent spaces, and thus is more flexible for modeling health-guided recommendation.

### 4.3 Fusion between preference and healthiness

We observe that existing health-guided food recommendation methods balance user’s food preference and healthiness via simple fusion. Therefore, we explore an effective non-linear fusion method between these two factors.

We extend the work in [27] where they integrate two types of representations from different neural networks via a knowledge transfer mechanism. Two knowledge transfer functions \(f_{p \rightarrow h}(\cdot)\) and \(f_{h \rightarrow p}(\cdot)\) are designed to allow the semantics of items in the preference and health graphs to influence each other. The knowledge transfer between \(\mathbf{v}^{(p)}_i\) and \(\mathbf{v}^{(h)}_i\) can be formulated as:

\[
\mathbf{v}^{(p)}_i = (1 - \alpha) \mathbf{v}^{(p)}_i + \alpha f_{h \rightarrow p}(\mathbf{v}^{(h)}_i)
\]
(15)

\[
\mathbf{v}^{(h)}_i = (1 - \alpha) \mathbf{v}^{(h)}_i + \alpha f_{p \rightarrow h}(\mathbf{v}^{(p)}_i)
\]
(16)

where \(\mathbf{v}^{(p)}_i\) and \(\mathbf{v}^{(h)}_i\) represent the augmented preference and health representations of item \(v_i\) and \(\alpha\) is the weight of the integration. The knowledge transfer networks \(f_{p \rightarrow h}(\cdot)\) and \(f_{h \rightarrow p}(\cdot)\) are implemented by fully-connected neural network layers. It allows useful knowledge to be extracted and transferred between different item semantics. In addition, we apply the back-transfer Zhu et al. [30] to further improve the quality of augmented representations:

\[
\mathbf{v}^{(p)}_i = (1 - \mu - \beta) \mathbf{v}^{(p)}_i + \mu f_{h \rightarrow p}(\mathbf{v}^{(h)}_i)
\]
(17)

\[
\mathbf{v}^{(h)}_i = (1 - \mu - \beta) \mathbf{v}^{(h)}_i + \mu f_{p \rightarrow h}(\mathbf{v}^{(p)}_i)
\]
(18)

where \(\mu\) and \(\beta\) are weights for the combination. Because the knowledge transfer between preference and health aspects is highly under-constrained, back-transfer serves as a cycle consistency constraint to force \(f_{h \rightarrow p}(\cdot)\) and \(f_{p \rightarrow h}(\cdot)\) to be inverse of each other.

### 4.4 Model optimization

For each node \(v \in \mathcal{V}\), we obtain its final L-order preference representation, denoted \(\mathbf{v}^{(p)}\), and its health representation, denoted \(\mathbf{v}^{(h)}\,\), from the preference and health graphs, respectively. To predict the probability \(\hat{y}_{uv}\), \(\mathbf{v}^{(p)}\) is fed into the prediction function \(f(\cdot)\) together with the final user representation \(\mathbf{u}\) learnt from Eq. (9): \(\hat{y}_{uv} = f(\mathbf{u}, \mathbf{v}^{(p)}) = \sigma(\mathbf{u}^T \mathbf{v}^{(p)})\), where \(\sigma\) is the sigmoid function. The loss function for the preference graph learning is defined as:

\[
L_p = \sum_{uv \in \mathcal{E}} \left( \sum_{i \in \mathcal{Y}_{uv}} J(\hat{y}_{uv}, \hat{y}_{uv}) - \sum_{i=1}^Q E_{i \rightarrow (p)} J(\hat{y}_{uv}, \hat{y}_{uv}) \right)
\]
(19)

where \(J\) is cross-entropy loss.
To make the computation more efficient, we use a negative sampling strategy during training. \( P \) is a negative sampling distribution over \( \mathcal{V} \) with \( y_{uv} = 0 \), and \( Q \) is the number of negative samples. \( P \) follows a uniform distribution. For the health graph, we do not use the user representation for training. Rather, we use a commonly-used graph-based loss, which concerns nearby nodes in graph to \( \mathcal{V} \) have similar representations:

\[
\mathcal{L}_h = -\log \left( \sigma \left( \mathbf{v}^T \mathbf{j}^b \right) \right) - \sum_{j=1}^{Q} \mathbb{E}_{v \sim \mathcal{P}(v)} \log \left( \sigma \left( -\mathbf{v}^T \mathbf{j}^b \right) \right)
\]

(20)

where \( j \) is a sampled nearby item, \( j' \) is a sampled negative (far-away) item, and \( \sigma \) is the sigmoid function.

To balance the different losses, the final loss

\[
\mathcal{L} = (1 - \eta) \mathcal{L}_p + \eta \mathcal{T}^U \mathcal{L}_h
\]

is a weighted linear combination of these two, with

\[
\mathcal{T}^U = \left| \{ u \in \mathcal{U} \} \right|
\]

Note that losses will be summed over all nodes across the batches and the weight determined by a scalar \( \eta > 0 \).

5 Experiments

Here, we evaluate our health-guided recommendation model and present its performance on two user–item interaction datasets derived from two different real-world food data collections. The code and dataset are publicly available at: https://github.com/DiyaLI916/recipe_recommendation. In this section, we aim to answer the following research questions:

- RQ1: How does our model perform compared with other state-of-the-art KG-based recommendation and recipe recommendation methods?
- RQ2: How do different components (i.e., propagation models, aggregation methods, and fusion mechanism) and component parameters (e.g., layers in GNNs) affect our model?
- RQ3: Can our model provide reasonable explanations about user preferences towards items with appropriate health guidance, and what are some of the typical error cases for our model?

5.1 Datasets

Most of previous research for recommending recipes relies on Web resources, e.g., Allrecipe, \(^2\) Cookpad, \(^3\) and Yummly, \(^4\) but they are not sufficient to develop advanced food recommender systems, which require large-scale user–item interaction datasets. A large-scale dataset with rich user-food interaction and multimodal content (e.g., ingredients, nutrition information and user reviews) is crucial for recipe recommendation, though there are only a few public large-scale datasets which comprise rich attributes for recommendation \([5,31]\).

Majumder et al. \([5]\) provides a user–item interaction dataset\(^5\) which is scraped from the Food.com website consisting of 180K+ recipes and 700K+ recipe reviews covering 18 years of user interactions from 2000 to 2018. The FoodKG \([6]\)\(^6\) knowledge graph is a large-scale KG that integrates recipes, food ontologies and nutritional data. It comprises about one million recipes (sourced from \([32]\)), 7.7 thousand nutrient records (sourced from the USDA: www.usda.gov), and 7.3 thousand types of food (sourced from \([33]\)). Overall, it has over 67 million triples. It is a great resource for KG-based recipe recommendation, and we therefore use FoodKG as the underlying food KG. Considering the scale of the user–item dataset and the direct mapping with the FoodKG \([6]\), we do recipe recommendation based on the Food.com dataset. We map the recipes in Food.com interaction dataset to the FoodKG, since over 500K recipes in the KG are from Food.com, and thus we construct the two graphs – preference and health graphs – for health-guided recipe recommendation.

To show the generality of our model, we construct another dataset from MyFitnessPal, \(^7\) which contains 587K+ food log records for 9K+ MyFitnessPal users from September 2014 through April 2015 \([7]\). Different from the direct mapping between Food.com and FoodKG, the food intake records in MyFitnessPal are not all recipes. For example, records can refer to snacks like “Quest Bar Cookies & Cream”. We therefore do approximate matching between the food items in MyFitnessPal and FoodKG by calculating the similarities of their titles and nutritional content. To increase the quality of the final user–item interaction datasets and the associated graphs, we leave out users having less than 5 records for food preferences, and infrequent items with frequency lower than 10. The statistics of the two preprocessed datasets are listed in Table 2.

For each dataset, we randomly select 90% of interaction history of each user to build the training set, and treat the remaining as the test set. We then randomly select 10% of interaction history from the training set as validation set for hyper-parameter tuning. For each record in the user–item interaction dataset, we treat it as a positive sample, and then we sample negative interactions for each user, indicating that the user has no interest in these items. The Food.com dataset has been used for recipe generation \([5,34]\) and the MyFitnessPal dataset has been used for dietary prediction \([7]\). To the best of our knowledge, we are the first to use these two datasets for KG-based recipe recommendation.

5.2 Experimental settings

5.2.1 Baselines

Existing works in food recommendation have very different task settings like recommendation via Q&A \([14]\), substituting food with healthier choices \([16]\), or recommending food to specific users \([18]\). However, we cannot directly compare our model with them due the differences in the task settings (e.g., with respect to \([14]\)) or unavailability of the source code (for \([16,18]\)). Instead, to demonstrate the effectiveness, we compare our proposed model with state-of-the-art KG-based recommendation methods and several competitive baselines as follows:

- **KGNN** \([19]\): This model is representative of propagation-based methods for KG-based recommendation. It focuses on representation enrichment of item KG by transforming the

<table>
<thead>
<tr>
<th>Table 2</th>
<th>Statistics Food.com and MyFitnessPal Datasets.</th>
</tr>
</thead>
<tbody>
<tr>
<td>User–Item Interaction</td>
<td>#users</td>
</tr>
<tr>
<td>Food.com</td>
<td>24,957</td>
</tr>
<tr>
<td>MyFitnessPal</td>
<td>9897</td>
</tr>
<tr>
<td>Knowledge Graphs</td>
<td>#entities</td>
</tr>
<tr>
<td>Food.com</td>
<td>213,746</td>
</tr>
<tr>
<td>MyFitnessPal</td>
<td>58,562</td>
</tr>
</tbody>
</table>


\(^3\) www.cookpad.com.


\(^6\) foodkg.github.io/.

\(^7\) www.kaggle.com/zvikinozadze/myfitnesspal-dataset.
KG into a user-specific weighted graph and then applying a GCN to compute the item embeddings. Note that the KG used in this work is simplified as a graph input for GCN.

- **KGNN-LS** [3]: As an extension of Wang et al. [19], this is a state-of-the-art model for the refinement of item representations. It additionally adds a label smoothness loss relying on the assumption that adjacent items in the KG are likely to have similar user relevance labels.

- **GCN**: This is a plain GCN model for item embeddings. To allow health-guided recommendation, FSA health ratings are incorporated as an important attribute during the propagation process. An additional loss to measure the healthiness aspect is considered by maximizing the health scores of predicted recommendations.

- **GAT**: This baseline has the same setting as the GCN baseline, but the propagation method uses GAT.

- **GCN + Post-filtering**: This baseline uses a GAT model for item embeddings. To be comparable with previous works in health-guided food recommendation, we adopt a straightforward post-filtering procedure [17] which re-weights the final item prediction probability \( \hat{y}_{uv} \) by health scores to allow health-guided recommendation. We set the re-weighted prediction probability \( \hat{y}_{uv} = \frac{\hat{y}_{uv} \cdot \sum_{h \in H}^N}{3N} \) (note that maximum nutrition score can be 3N). The post-filtering method is widely used in food recommendation to achieve health goals.

- **Content-based filtering + Post-filtering**: This baseline is chosen as a traditional non-graph-based food recommendation model. Content-based filtering uses recipe attributes to recommend other recipes similar to what the user likes. To address the healthiness aspect, the recommendation results are re-ranked via the same post-filtering procedure used in the GCN + Post-filtering baseline.

- **GCN + GCN**: This combined model can be treated as an ablated version of our proposed model by replacing GAT with GCN for the health graph.

- **GCN + GAT w/o back-transfer**: This combination is an ablated version of our model without back-transfer during knowledge fusion by removing Eqs. (17), (18).

We chose the above baselines with regards to four aspects: (i). Our main focus is on the health aspect in KG-based food recommendation. However, to see how general domain KG-based recommendation methods perform on our task, we use two state-of-the-art KG-based recommendation methods, namely KGNN and KGNN-LS. (ii). We study two popular GNN baselines modeling the user preference and health aspects in one GNN, namely GCN and GAT. We compare with these models to demonstrate that separately modeling the two aspects (preferences and health) as done in our approach can yield better recommendations. Incidentally, both the GCN and GAT baselines are also KG-based since the input graph structures are generated from KGs. (iii). To study the effect of post-processing, as done in some previous works, we include a post-filtering method, namely GAT + Post-filtering, as a competitive baseline representative of previous work on health-guided recipe recommendation. (iv). We include two baselines for our ablation study, namely GCN + GCN and GCN + GAT w/o back-transfer. These show the effectiveness of our proposed GCN (for preference graph) plus GAT (for health graph) approach. As such these baselines are representative of both KG and non-KG based state-of-the-art recommendation approaches.

### Table 3

<table>
<thead>
<tr>
<th>Hyper-parameters settings.</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Food.com</strong></td>
<td></td>
</tr>
<tr>
<td>( \lambda )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \delta )</td>
<td>0.5</td>
</tr>
<tr>
<td>( N )</td>
<td>5</td>
</tr>
<tr>
<td>( d )</td>
<td>128</td>
</tr>
<tr>
<td>( L )</td>
<td>2</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>0.5</td>
</tr>
<tr>
<td>( \mu, \beta )</td>
<td>0.1</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.5</td>
</tr>
<tr>
<td><strong>MyfitnessPal</strong></td>
<td></td>
</tr>
<tr>
<td>( \lambda )</td>
<td>0.3</td>
</tr>
<tr>
<td>( \delta )</td>
<td>0.5</td>
</tr>
<tr>
<td>( N )</td>
<td>5</td>
</tr>
<tr>
<td>( d )</td>
<td>128</td>
</tr>
<tr>
<td>( L )</td>
<td>2</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>0.4</td>
</tr>
<tr>
<td>( \mu, \beta )</td>
<td>0.1</td>
</tr>
<tr>
<td>( \eta )</td>
<td>0.3</td>
</tr>
</tbody>
</table>

#### 5.2.2 Evaluation metrics

We evaluate our method via three metrics described below:

- **Click-through rate (CTR) prediction** [35]: We apply our model to predict each user-item pair in the test set including positive items and randomly selected negative items. We adopt AUC (Area Under The Curve) as the evaluation metric in CTR prediction.

- **Top-K recommendation** [36]: We use the trained model to select K items with highest prediction score for each user in the test set, and choose Recall@K (or R@K) to evaluate the recommended results.

- **Health-guided top-K recommendation**: Additionally, in order to measure the healthiness of the top-K recommendations, we design a revised top-K score where the health scores of the top-K recommendations are considered. The health-revised recall at top-K (HR@K) is calculated as:

\[
HR@K = \frac{\sum_{v \in \mathcal{V}^K} h_v / |\mathcal{R}|}{N},
\]

where \( \mathcal{V}^K \) denotes the collection of relevant recommended items at top-K, \( h_v \) is the health score of the relevant recommended item \( v \), \( |\mathcal{R}| \) is the average health score of all relevant items, and \( N \) is the total number of relevant items.

The motivation for designing health-revised recall is that we want to measure not only the proportion of relevant items found in the top-K recommendations, but also the healthiness degree of the relevant items in the top-K compared to all relevant items. Essentially, HR@K is a variant of Recall@K weighted by recipe health scores. We have HR@K = R@K = 1 if all relevant items are hit in the top-K recommendations since \( \sum_{v \in \mathcal{V}^K} h_v = N \cdot \bar{h} \).

#### 5.2.3 Parameter settings

We implemented our model in Tensorflow 1.12.0 and all experiments are conducted an Intel i7-2700K CPU and an Nvidia Titan Xp GPU with 16 GB of memory. For each dataset, the ratio of training, validation, and test set is 8:1:1. Each experiment is repeated 5 times, and the average performance is reported. All trainable parameters are optimized using Adam [37], where the batch size is fixed at 256. We set the dimension the item embedding \( d \) as 128.

The main hyper-parameters for our model are listed in Table 3. The random walk restart probability is 0.2 for constructing subgraphs. \( \lambda \) is the edge weight for preference graph and \( \delta \) is the
edge weight for health graph construction. \(N\) denotes number of sampled neighbors for each entity, \(d\) is the dimension of user and item embeddings and \(L\) represents number of GCN and GAT layers. \(\alpha\) is knowledge transfer weight; \(\mu\) and \(\beta\) are back transfer weights. Lastly, \(\eta\) is the scalar for the weighted loss function. Hyper-parameter settings were determined by optimizing \(R@10\) on the corresponding validation sets.

5.3 Performance comparison (RQ1)

We first examine how our model performs compared with other state-of-the-art KG-based and recipe recommendation methods. The comparative performance results are presented in Table 4. We make the following observations:

- Existing KG-based recommender systems (KGNN, KGNN-LS) give reasonable recommendations to some degree. However, the effectiveness of these methods is limited as they mostly model the message propagation over different edge relations, whereas the edge relations in FoodKG are scarce.
- Compared to baselines modeling the user preference and health aspects together in one GNN (GCN and GAT), our proposed model outperforms them by a huge margin. It reveals that separately training these two aspects in different graphs, then fusing them with smart knowledge transfer mechanism can better capture different item semantics, resulting in better recommendation.
- The GAT with post-filtering procedure is not very effective. It is hard to optimize the trade-off between user preference and healthiness since the achievement in health-revised recall is not particularly high and results in poor recommendation accuracy.
- Content-based filtering with post-filtering performs the worst, highlighting the power of KG-based recommendation. Furthermore, comparing the performance of non-graph model (content-based post-filtering) and graph model (GAT with post-filtering) on the Food.com and MyFitnessPal datasets, we observe that the graph-based model can achieve better performance when items are largely mapped into external KG and thus lead to better graph structures.
- The AUC score on Food.com is much higher than MyFitnessPal across the board, which makes sense as we perform approximately matching to build the user–item interaction dataset for MyFitnessPal. There might be some inappropriate mappings between MyFitnessPal and FoodKG; more effective measures can be explored to improve the quality of the dataset in the future. In contrast, the recall scores at the top-50 in MyFitnessPal is greater than Food.com, since the data scale of MyFitnessPal is small and the retrieval space is largely reduced.

With regards to the health-revised recall \(HR@K\), our model consistently yields the best performance by a significant margin on all the datasets (along with high scores in AUC and regular recall too). This demonstrates that our proposed model can recommend “tasty” as well as “healthy” food to users.

5.4 Component analysis (RQ2)

We first examine how the different components affect our model.\(^8\) To answer this question, we have two ablated versions of the propagation method and fusion mechanism. As shown in Table 4, compared to our final model (Ours; last row), the GCN + GNN baseline gets relatively high scores in AUC and Recall@K, but has worse performance for HR@K. GAT plays an important role in the propagation process over the health graph, since the relation between recipes in the health graph is implicit and GAT can automatically assign weights to neighbors via the attention mechanism. The GCN + GAT w/o back transfer model obtains the highest scores in AUC and Recall@K. One potential reason might be that the constraint between the user preference and food health is relaxed without the back-transfer, thus encouraging the item representation learning towards the user preference. Compared to this ablated version, our final model gets the highest HR@K scores with a very slight sacrifice on the traditional evaluation metrics. As our goal is to do health-guided recommendation, our fusion-based approach has a much higher HR@K score. It is reasonable to accept a minor decrease in the user preference aspect, but with a better HR@K score.

The ablation study also justifies our choice of GCN for the preference graph and GAT for the health graph. Using GCN for the preference graph allows for non-parametric weights to be assigned to different neighbors to make the propagation relation-specific. The GCN model is further improved by adding a user-specific weight (achieved by Eq. (9)). For learning in health graph, the relation between items becomes implicit since it is regulated by health scores. The assumption of “a user will have distinct preferences for different relations” does not hold in this case. Therefore, we use GAT instead of GCN for information aggregation. This choice is proved to be effective in the ablation study referring to the GCN + GCN baseline.

Furthermore, to analyze the sensitivity of our proposed model to the number of GNN propagation layers \(L\), we vary \(L\) from 1 to 3 and try different combinations for GCN and GAT. The results are listed in Table 5. GCN-2 + GAT-2 achieves the best performance on AUC and Recall@K, indicating that increasing the depth of GNNs can boost performance. The results in Table 2 for

---

\(^8\) We have conducted experiments on different aggregators for GCN (Eqs. (6), (7), (8)), but the performance differences are minor; thus, the results are not reported here.
Fig. 2. Illustration of a case study from the Food.com dataset.

Table 5
Effect of GNN Propagation Layer Numbers (L).

<table>
<thead>
<tr>
<th>Model</th>
<th>Food.com</th>
<th>MyFitnessPal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>AUC</td>
<td>R@10</td>
</tr>
<tr>
<td>GCN-1 + GAT-1</td>
<td>0.712</td>
<td>0.027</td>
</tr>
<tr>
<td>GCN-1 + GAT-2</td>
<td>0.714</td>
<td>0.027</td>
</tr>
<tr>
<td>GCN-2 + GAT-1</td>
<td>0.724</td>
<td>0.034</td>
</tr>
<tr>
<td>GCN-2 + GAT-2</td>
<td>0.725</td>
<td>0.035</td>
</tr>
<tr>
<td>GCN-3 + GAT-1</td>
<td>0.720</td>
<td>0.031</td>
</tr>
<tr>
<td>GCN-3 + GAT-2</td>
<td>0.718</td>
<td>0.029</td>
</tr>
</tbody>
</table>

5.5 Case study: Explanation of recommendation and error analysis (RQ3)

We now present a case study to showcase that our model can recommend healthy and appropriate food items. One advantage of KG-based recommender system is that it makes recommendation explainable to some extent by offering reasoning along paths in the graphs. To this end, we randomly select one user–item pair \((u, v)\) from the test set.

As shown in Fig. 2, for the user \(u\), who loves Classic New York Cheesecake \((v)\), Rosie’s New York Cheesecake Brownies \((v_5)\) with desirable health score (amber means neutral) is recommended for him based on our model. Without considering the healthiness, general recommendation suggests items like \(v_1, v_2, v_3\), and \(v_4\) which are different types of cheesecakes, since they are closer to \(v\) in the preference graph, whereas they are unhealthy recording to the FSA ratings (red denotes unhealthy). Taking the health aspect into consideration, our model jointly learns item representations via the health graph where Rosie’s New York Cheesecake Brownies is close to Classic New York Cheesecake. The item \(v\), Classic New York Cheesecake and its neighbors in the two graphs are illustrated in Fig. 2. Their health scores are represented in color. After the knowledge transfer process, the representation of item \(v\) is further enriched by item \(v_5\). In the end, \(v_5\) gets a higher prediction score for recommending to \(u\). Table 6 summarizes these results; it shows the top-5 recommendations both without...
and with health aspect ranking for the given query v: Classic New York Cheesecake. We can clearly observe that when we do not take the health graph into account, the recommended items are mostly unhealthy. On the other hand, taking health graph into account, our approach can recommend a healthier alternative. It also suggests an even more healthy sugarless alternative, but it is ranked lower given the user preferences. Our model can thus recommend food that both meets the user’s preferences and also considers the healthiness, offering potential explanations as to why a healthier food is suggested by checking against the health graph generated from the KG.

Moreover, we aim to examine the quality of the recommendation results generated from different types of systems. The top-5 recommendation results generated from the content-based post-filtering baseline, the KGNN-LS baseline and our health-aware KG-based model are listed in Table 7. The content-based filtering approach tends to suggest simple recipes covering key words in the user preferences and it is prone to producing redundant results. The KGNN-LS method can provide relevant and diversified results compared to the non-graph model by leveraging semantic and structural information from the external KG. Our model shows evenly matched results with KGNN-LS, while yielding healthier recommendations (see the healthiness color codes in the bullets). As shown in Table 7, in contrast to the unhealthy Carl’s mango margarita suggested by KGNN-LS, a healthier beverage peach mango tea sangria is selected by our system with a higher rank.

To further examine the interpretability of these recommended results, we selected some matching pairs (a, b) where a is an item from the user’s preference, and b is the recommended item with regards to a. For instance, \((\text{tomato phyllo pizza}, \text{margarita pizza})\) marked in blue and \((\text{lemon raspberry tiramisu}, \text{cranberry tiramisu (gluten-free)})\) marked in brown are two matching pairs in Table 7. By mapping these pairs back to the food KG, we can find relevant KG triples that connect them. We list some of the representative triples \(t_1, \ldots, t_9\) in Table 7. The triples that are related to the recommended recipes are listed along with that item in Table 7. The relevant KG triples offer some intuition about how the graphs are constructed from the original KG and thus implicitly helping item representation learning. For instance, tomato phyllo pizza is linked with margarita pizza during our graph construction procedure. According to KG triples \(t_1\)–\(t_5\), tomato is the common ingredients in tomato phyllo pizza and margarita pizza. Fat free mozzarella cheese is a subclass of mozzarella cheese according to the food KG, thus mozzarella cheese is another common ingredient in these two pizzas. The item margarita pizza remains as a neighbor of tomato phyllo pizza both in preference and health graphs after random walks and learns similar representations. The margarita pizza is finally recommended for tomato phyllo pizza. Similarly, lemon raspberry tiramisu and cranberry tiramisu (gluten-free) are close; one clue we can draw from the KG is that this is due to the fact that cranberry can be substituted for raspberry. These examples illustrate the power of KGs, not only in improving the quality of retrieval, but also in interpretation of the recommendation results.

To gain more insight, we analyze error cases and show two typical cases in Table 8. Case 1 highlights the limitation of using exact match when computing the recall score metric, whereas Case 2 shows the issue of highly diversified user preferences. For Case 1, the top-5 recommended results generated from our model obtain a recall score of zero when compared with ground truth since we the recommended items do not have an exact match with the ground truth recipe ids. However, we can see that grilled salmon and creamy Cajun chicken pasta with bacon can be treated as approximate matches for honey ginger grilled salmon and creamy Cajun chicken pasta in ground truth. The recommended recipes can thus obtain a higher match score with the ground truth if we were to measure the similarities of recipe titles. We have observed that many recipes in food.com have the same titles with minor differences in their ingredients; and even users cannot tell the differences apart. This indicates that we can achieve considerable user satisfaction in practice. As for Case 2, we can see that the user items (recipes) in the ground truth are quite varied. When the user preferences are so diversified, it is hard for our model to capture user intent and recommend appropriate dishes. Thus, the top-5 results have no overlap with any items in the ground truth. Moreover, we notice that there are two duplicate recommendations, properly prepared spaghetti squash. These are distinct recipes with unique recipe ids, but with the same title. This suggests a re-ranking strategy to avoid redundancy in final recommendation.

### 6 Conclusion

We note that there are only a few food recommendation works that utilize knowledge graphs, and that the health aspect is crucial for good dietary choice. To fill this gap, we propose a framework to do health-based recipe recommendation over KGs. Our novel contribution is thus the application of KG-based recommendation in a real-world recipe application that also considers healthiness scores in a effective joint model. To our knowledge this is the first work on the important task of KG-based recipe recommendation. We jointly train two types of

---

9 Though it may sound strange, there is in fact a recipe called “yes, Virginia there is a great meatloaf” – [https://www.food.com/recipe/yes-virginia-there-is-a-great-meatloaf-54257](https://www.food.com/recipe/yes-virginia-there-is-a-great-meatloaf-54257).
A knowledge transfer scheme is further adopted to fuse the two important aspects, thus achieving the goal of recommending food that is both “tasty” and “healthy”. That is, since we explicitly consider a recipe’s health aspects during training, we are thus able to recommend healthier food. That is, if there are similar recipes, recommendation based purely on preference may return an unhealthy one, but our method will recommend the more healthy one (but still similar). We use a large real-world KG, and experiments on two real-world datasets demonstrate the effectiveness and interpretability of our model. Compared to ad-hoc or post-hoc incorporation of health-aspects in previous works, our approach considers health scores to be equally important compared to user preferences, which is more systematic and is handled more effectively in our joint end-to-end model.

This work opens up several directions for future research. Firstly, we can improve the recommendation scores by considering approximate matching instead of exact matching of the recipe titles, as highlighted in the error case analysis. Likewise, we can re-rank results to avoid duplicate titles (but distinct recipes). Future work also needs to improve on the case where the user preferences are very diverse, perhaps by grouping them into similar clusters, and by considering user intent. The current healthiness based recommender results are derived from the items without considering user profile information such as dietary restrictions or other constraints. There are also many other useful modalities in recipes that can be incorporated into food recommendation such as the recipe images, cooking instructions, user defined tags, and so on. In the future, we plan to explore more recipe modalities and to do personalized health-guided food recommendation by focusing on the user profile for specific health goals.

CRediT authorship contribution statement

**Diya Li:** Conceptualization, Methodology, Software, Writing – original draft. **Mohammed J. Zaki:** Conceptualization, Methodology, Supervision, Writing – review & editing. **Ching-hua Chen:** Methodology, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Diya Li reports statistical analysis and writing assistance were provided by IBM Research.

Acknowledgments

This work is supported by IBM Research AI through the AI Horizons Network (under the RPI-IBM HEALS Project).

References


Table 8

<table>
<thead>
<tr>
<th>Ground truth</th>
<th>Top-5 recommended results considered healthiness</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Honey ginger grilled salmon</td>
<td>• Chicken noodle soup</td>
</tr>
<tr>
<td>• Apricot honey grilled chicken</td>
<td>• Cauliflower steak with olive relish</td>
</tr>
<tr>
<td>• Crock pot whole chicken</td>
<td>• Grilled salmon</td>
</tr>
<tr>
<td>• Chocolate toffee candy cookies saltine candy</td>
<td>• Chicken pear salad</td>
</tr>
<tr>
<td>• Fannie farmer’s classic baked macaroni cheese</td>
<td>• Creamy cajun chicken pasta with bacon</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Case 1</th>
<th>Case 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Yes, Virginia there is a great meatloaf</td>
<td>• Oven crisp chicken wings</td>
</tr>
<tr>
<td>• The best chili you will ever taste</td>
<td>• Properly prepared spaghetti squash</td>
</tr>
<tr>
<td>• Cheesy shrimp grits casserole</td>
<td>• Properly prepared spaghetti squash</td>
</tr>
<tr>
<td>• Sausage gravy</td>
<td>• Olive garden copycat zuppa toscana</td>
</tr>
<tr>
<td>• Do at home onion rings</td>
<td>• Old fashioned vegetable beef soup</td>
</tr>
<tr>
<td>• Southern sweet iced tea</td>
<td></td>
</tr>
</tbody>
</table>


