Healthy Recipe Recommendation using Nutrition and Ratings Models

Yew Siang Tang, Anita Hanzhi Zheng, Nicholas Lai
Stanford University

Abstract

We propose a three-part algorithm to provide healthier and tastier recipe alternatives given a person's food preferences. The first part consists of developing a nutritional model using linear regression to understand the overall nutritional content of a recipe given its ingredients. The second part consists of modelling a user's rating scores using a graph neural network (GNN) on ingredient-recipe and recipe-user bipartite graphs. We combine these two models to enable us to evaluate the healthiness and tastiness of novel recipes according to users' preferences. We show that our GNN approach towards the ratings model achieves strong performance and provide compelling qualitative results of our recommended recipes.

Introduction

The problem of how to cook good food and make it nutritious has existed for as long as humans have existed. For thousands of years, the main ways people learned about great dishes and recipes was either through trial and error or word of mouth, and neither method took much direct consideration of a dish's nutrition value. Today's technological advances have made it possible to leverage other people's food preferences to not only suggest dishes one might like, but also craft them in such a way that they satisfy the nutritional requirements people need on a daily basis. As this problem can be expressed as a graph between users, recipes, and ingredients, we think we can use modern techniques in machine learning and graph theory to suggest better recipes than many standard systems for rating recipes.

In particular, we have decided to focus mostly on the relationships between ingredients, recipes, and users to create our recommendation system, as we think these are the most pertinent parts of a recipe with regards to recommending healthy and tasty recipes. We have also decided to separate modeling nutrition of recipes from modeling tastiness of recipes, as nutritional data in general tends to be less subjective and easier to reason about than rating data. Finally, we have chosen to emphasize comprehensibility and flexibility of our models over exhaustiveness, as we think that ultimately anyone should be able to use our recommendations, if not build a similar system themselves.

With this framework in mind, we can create suitable nutrition and ratings models which properly leverage relationships between users, recipes and recipe ingredients, the details of which are explained more fully under Method.

Related Work

Flavor Network and the Principles of Food Pairing

Flavor networks (Ahn et al. 2011) is a less technical yet still rigorous paper which presents fairly standard statistical analysis on the graph of ingredient networks. The formula $N_s(R) = \frac{2}{n_R(n_R-1)} \sum_{i,j \in R, i \neq j} |C_i \cap C_j|$ is used to represent the mean number of shared compounds, and base many measures used in the paper on this metric, including contribution and authenticity. Evaluating this metric with a z-test, the paper has found that ingredient pairs in North America tend to share more compounds while East Asian cuisines tend to share fewer compounds than expected by chance, and the distribution of N_S values forms an approximately normal distribution.

Graphs are also presented which show that certain ingredients have especially high usage, such as garlic, scallion, tomato, and olive oil. The number of ingredients per recipe forms an approximately beta distribution with respect to P(s), and the rank and frequency distribution per cuisines look remarkably similar to a Zipf-Mandelbrot curve.

Recipe Recommendation Using Ingredient Networks

Recipe Recommendation Using Ingredient Networks by Teng, Lin, and Adamic (Teng, Lin, and Adamic 2012) parses the unstructured text of recipes and user reviews on Allrecipes.com, and constructs two types of networks that reflect different relationships between ingredients, so as to capture users' knowledge about how to combine ingredients.

The first type of network, namely the complement network, captures which ingredients tend to co-occur frequently using pointwise mutual information, defined as $PMI(a, b) = \log \frac{p(a,b)}{p(a)p(b)}$. They show that the network is largely composed of two large communities, savory and sweet, and that the maximum PMI of a recipe is very slightly positively correlated with its average rating.

Graph	Node Count	Edge Count	Average clustering coefficient	Average degree
Ratings graph	51,987	133,459	N/A	4.55 (user out-degree) 5.88 (recipe in-degree)
Ingredients graph	10,874	269,962	0.868	49.65

Table 1: Statistics for ratings and ingredients graphs

The second type of network, namely the substitute network derived from user generated suggestions for modifications, is a weighted directed graph where the weight of edge from a to b denotes the proportion of substitutions of ingredient a that suggest ingredient b. It can be decomposed into many communities of functionally equivalent ingredients and captures users' preference for healthier variants of a recipe. Teng et al. demonstrate that the substitute network encodes users' ingredient preference.

With these networks, Teng et al. also conduct experiments using stochastic gradient boosting trees, showing that recipe ratings can be predicted by features from combinations of ingredient networks and nutrition information with accuracy 0.792, with most predictive power coming from the ingredient networks.

Exploiting Food Choice Biases for Healthier Recipe Recommendation

Elsweiler et al. (Elsweiler, Trattner, and Harvey 2017) want to improve the recipe recommendation / ranking process to include the health information of the dishes so that users are given tasty and healthy recipes. In this problem setting, a user has a desired dish on their mind and finds recipes on online recipe recommendation systems. This problem is challenging because popular and highly-rated recipes correlate positively with high fat, sugar and calorie content, i.e. tasty food tends to be nutritionally poorer.

To recommend dishes that balance tastiness with healthiness, the authors substitute a user's desired recipe with similar recipes of higher nutritional content and at least an equivalent rating in the recipes database, using cosine similarity to filter out dissimilar recipes and recipes with lower ratings. In addition to developing a recipe substitution system, the authors study the ability of people to judge fat content of recipes and their biases in their selection of recipes, finding that people implicitly prefer fatty food but yet are not able to distinguish the fat content between foods. In their findings, they describe how food titles and images can bias people in their perception of fat content and simple low-level food image features can be highly predictive of food preferences.

General Limitations of Existing Methods

Based on the previous papers, we have found that these methods either analyze but do not recommend recipes, disregard nutritional information, are difficult to comprehend by users or are constrained to suggesting only recipes that are within databases. We believe that we can use our knowledge about machine learning on graphs to allow us to create a better recipe model which accounts for both nutrition and taste.

In order to do this, we made a few key choices to improve the model. To start, we have decided to assume that recipes are bags of ingredients, to improve one's encoding of, say, dietary restrictions or personal preferences. We have also decided not to focus on cultural characteristics of food, as we have seen that past papers have undertaken such analysis with data heavily biased towards European and North American dishes. Furthermore, we have included a means to generate new recipe "substitutes" as part of the algorithm, as our project intends to find healthier alternatives to recipes currently existing in cookbooks. Above all, we made these choices so our output model should be understandable to laypeople, even if the means of obtaining such a model is somewhat technical.

Dataset

Our dataset, from Kaggle, consists of about 50,000 recipes scraped from allrecipes.com, with each recipe containing the name and ID on the website, the average rating over all the reviews on the website, an accompanying image on the website, the ingredients used for the recipe, the cooking directions, and the nutrition content; most relevant for our use case will be the dish name, the recipe's ingredients, the review data, and the nutrition. The nutrition data consists of a dictionary corresponding to the standard nutrition content tables found to the side of most American food products, and the review data consists of user IDs with review score and timestamp per dish.

Preprocessing Ingredient Lists

The following are examples of user-provided ingredients:
[sauerkraut drained, Granny Smith apples sliced, large onion, caraway seeds, apple cider divided, brown sugar, Rub:, Thai seasoning, salt, garlic powder, ground black pepper, boneless pork loin roast]
[kielbasa (Polish) sausage (such as Hillshire Farm®), grape jelly, bottles barbeque sauce]

As the ingredients are specified differently by online users, there is significant noise. For example, both *'large onion'* and *'sliced onion'* are different descriptions used to refer to the same ingredient *'onion'*. In order to make use of the recipes, we need to extract only the relevant ingredient information from noisy user-provided ingredient lists. We used a combination of heuristics and Natural Language Processing tools to trim ingredient strings to their fundamental ingredients.

First, we removed unrelated characters and special symbols such as : and ®. Then, we used regular expressions to remove bracketed comments so that *'kielbasa (Polish) sausage (such as Hillshire Farm®)'* becomes *'kielbasa sausage'*. Finally, we made use of the Python Natural Language Toolkit (NLTK) library to filter words according to high-level con-

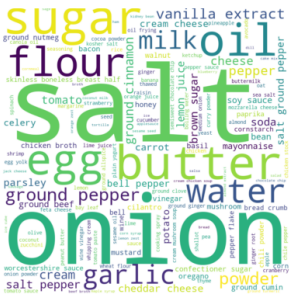

Figure 1: Word cloud of ingredient counts

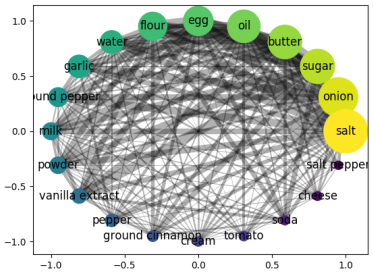

Figure 2: Ingredient co-occurence graph

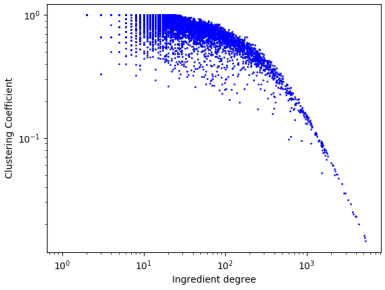

Figure 3: Ingredient clustering coefficient vs degree

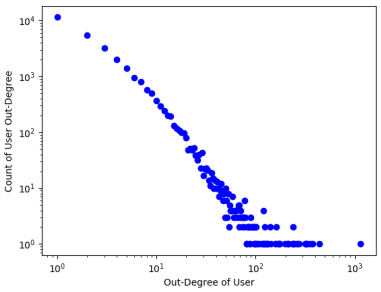

Figure 4: User degree

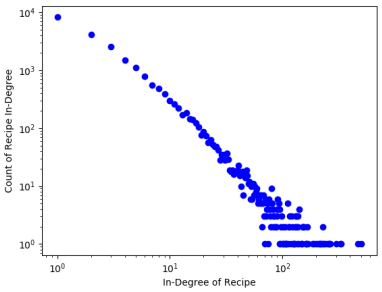

Figure 5: Recipe degree

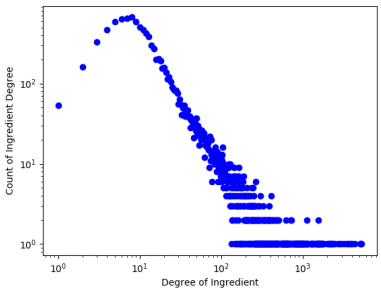

Figure 6: Ingredient degree

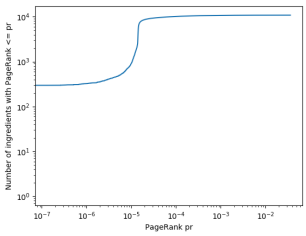

Figure 7: Ingredients PageRank cumulative distribution

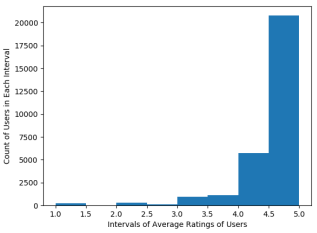

Figure 8: User average rating

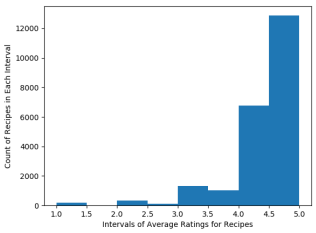

Figure 9: Recipe average rating

cepts. Specifically, we use a Part-Of-Speech (POS) tagger and keep only nouns. Therefore, *'Granny Smith apples sliced'* becomes *'Granny Smith apples'*. This greatly increases the accuracy of our ingredient retrieval and allows us to construct a high quality ingredients vocabulary as seen in Fig 1. Nevertheless, we can still improve further on this step to properly express tokens such as *'ounce club soda'* as *'club soda'*.

Data Statistics

For our initial analysis, we constructed a ratings (also recipe-user) graph which is a bipartite graph with users and recipes as nodes in different partitions, and directed weighted edges representing the ratings that users give for recipes. Additionally, we constructed an ingredients complement graph which is an undirected graph with ingredients as nodes, and edges representing the PMI between two ingredients as defined in Recipe Recommendation Using Ingredient Networks. Note that the ingredients complement graph is only for analysis and not used in the subsequent GNN algorithm. We generated a large number of different data statistics for each graph, and we present some of the more meaningful ones here. Some of the basic ones are shown in Table 1.

Our initial analysis consisted of creating degree distributions for the user data, recipe data, and ingredient data, as captured in the ratings graph and the ingredients complement graph (see Figure 4, 5, 6).

Based on an initial analysis, it seems that the ingredient complement graph may have an unusual data distribution, as one would expect the degree distribution to be approximately linear on a log-log distribution, as was the case for users and recipes (and most distributions). However, this might have been due to different sources of error: we could have allowed for false positives or negatives in our preprocessing, the ingredient lists may be an incomplete representation of

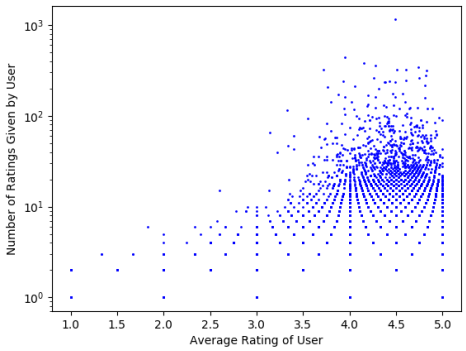

Figure 10: User average rating vs number of ratings

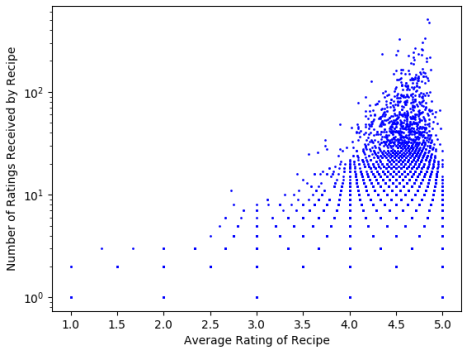

Figure 11: Recipe average rating vs number of ratings

Figure 12: Word cloud of inferred ingredients' carbohydrates content

Figure 13: Word cloud of inferred ingredients' protein content

a recipe in real life, or we may be using ingredient data from people who have biases towards certain ingredients, particularly Western ones. As we'll discuss later, having an unobjective dataset presents a host of problems which make drawing conclusions hard. In that context, statistics such as the average clustering coefficient and average degree for ingredients may not mean much until we take a closer look at our data pipeline.

To better understand the ingredients graph, we created a ingredient co-occurence graph (Figure 2), which is the top 20 ingredients over all recipes in terms of total co-occurence with other ingredients. The radius of each circle stands for total co-occurences with other ingredients, and the width of any edge stands for the raw count of co-occurences between the two ingredients. This figure demonstrates that our ingredients graph is mostly making sense for the common ingredients, since most of the thick connections in the figure make sense and the common ingredients on the figure are actually common in life. We then plotted the relationship between clustering coefficient and ingredient degree (Figure 3), which behaves as expected by showing the decline of clustering coefficients as degree increases. Furthermore, we computed the PageRank for each ingredient and plotted the cumulative distribution of PageRank (7). This shows that most ingredients have similarly small PageRank, while a small number have much larger PageRank, which further implies that ingredient PageRanks approximately follow a power law distribution.

For the ratings graph, we wanted to further understand distribution of user/recipe's average ratings. So we plotted the distribution of each user's average ratings (Figure 8), as well as the distribution of each recipe's average ratings (Figure 9). We then plotted the relationship between user/recipe's average rating and the number of ratings it has given/received (see 10, 11). As is shown by these diagrams, most users tend to give high ratings to recipes, and most recipes tend to receive high ratings. Further, low average ratings tend to only occur when a user/recipe has a very limited number of ratings.

This matches our intuition that a user is usually more likely to give positive ratings than negative ones, but this tendency might cause some class imbalance issues in our prediction task because the dataset is skewed towards high ratings.

Problem Statement

In this work, we want to make *novel* recipe recommendations that are both *healthy* and *tasty*. We define a recipe by its list of ingredients and it is novel if the recipe is not found within our dataset. A recipe's healthiness can be determined by its amount of healthy vs unhealthy nutrients (such as proteins vs sodium) while its tastiness can be approximated by the average ratings provided by a large number of users. However, because we are interested in making *novel* recipe recommendations where these recipes are out of our dataset, we cannot simply retrieve the nutrition content and ratings of these recipes. Consequently, we have to infer both the healthiness and tastiness of novel recipes through nutrition and ratings models that compute the nutritional content of a recipe and the rating that a user would give the recipe respectively. Then, with these models, we are able to make good healthy recipe recommendations by evaluating novel recipes attained through different combinations of ingredients and optimizing for a healthiness and tastiness score.

Method

Nutrition Model

In order to evaluate whether a recipe R is healthy, we need to know the nutritional content $H(R)$ of the ingredients within each recipe. While recipe datasets might provide an aggregate nutritional content, they do not reveal the individual components contributed by each ingredient. Therefore, if the ingredients were changed, the nutritional content will not longer be valid. Even if we used a known database of ingredients' nutritional content to facilitate the change of ingredients, we will also need to know the mass of ingredients in the recipe. Since we know neither the ingredient mass nor individual ingredient nutrition content, we cannot compute new recipes' nutritional content.

We propose to solve the above problem by approximating $H(R)$ with our nutrition model $\hat{H}(R) = Rx$, where $R \in \mathbb{R}^{N_R \times N_I}$ is our full recipe sparse matrix where the entry R_{ij} indicates whether the ingredient I_j is present in the recipe R_i, and $x \in \mathbb{R}^{N_I \times h}$ is the vector that represents the h nutritional contents of an ingredient. As x should be non-negative in all components, we cannot simply solve the linear equation $Rx = y$ as it can lead to negative x components. Therefore, we solve the following constrained optimization problem,

$$\begin{aligned} &\arg\min_x \ \frac{1}{2}||Rx - y||^2 \\ &\text{subject to } x \geq 0 \end{aligned} \tag{1}$$

where $y \in \mathbb{R}^{N_I \times h}$ is the actual nutrition content of the recipe. To evaluate the nutritional content of a candidate recipe $R_i \in \mathbb{R}^{N_I}$, we can simply return the dot product $\hat{H}(R_i) = R_i^T x$.

Note that in this linear formulation, each ingredient is assumed to have the same nutritional content over all the different recipes. While this assumption is not true since different recipes use different amounts of ingredients, it is still a sound and useful assumption since it finds the most likely nutritional content given all the recipes. It is reasonable to assume that the nutritional contributions of broccoli and pasta in different recipes are unlikely to differ substantially. Additionally, it will allow us to tackle the noise inherent in the nutritional content provided by recipe databases.

Ratings Model

The ratings model predicts a user's rating on a recipe when given the history of the user's past ratings and the ratings of all other users. If the recipe is within our database and has been rated by other users before, then it corresponds to the standard recommendation systems that commonly use collaborative filtering algorithms. However, because we want to recommend novel recipes that are not within the database, these algorithms will fall short. To tackle this novel recipe ratings prediction task, we want to create a ratings model with several desiderata:

- The model should exploit the graph structure between ingredients, recipes and users to have a deeper understanding of the relationships. Specifically, there are two bipartite graphs, an ingredient-recipe graph where each recipe is linked to its component ingredients and a recipe-user graph where each edge represents the rating that the user have for the recipe.
- There should not be any retraining when there are new users or new recipes, because a novel recipe recommendation algorithm will likely encounter many new users and new recipes, and retraining in order to perform inference on them would make the application prohibitively slow and expensive.

Graph Neural Network Given the above desiderata, we design a graph neural network model based on a modification of the GraphSAGE algorithm on top of the ingredient-recipe and recipe-user bipartite graphs (as seen in the left of Figure 14) towards the problem of novel recipe rating prediction. Formally, let u_v, r_v, $s_{u,v}$ denote the user v, recipe v and the rating that user u gives for recipe v. We have a target user u_* and a target recipe r_*, and we would like to infer s_{u_*,r_*}, the rating that user u_* has for recipe r_*. There are three stages in the forward pass of the algorithm and we shall describe the procedure and training using Figure 14:

Stage I Recipe vectors (2nd from left): We want to collect information about the recipes that our target user u_* have rated before in order to understand user u_*'s recipe and ingredient preferences. Specifically, we aggregate ingredient embeddings information to understand what constitute the recipes, and we collect average rating information about the recipes and the target user's rating information about the recipes to understand how the user's preferences might differ from majority preferences. The resulting recipe vector $\overrightarrow{r}$ encapsulates this information. For $v \in \mathcal{N}_{RU}(u_*)$, $v \neq r_*$,

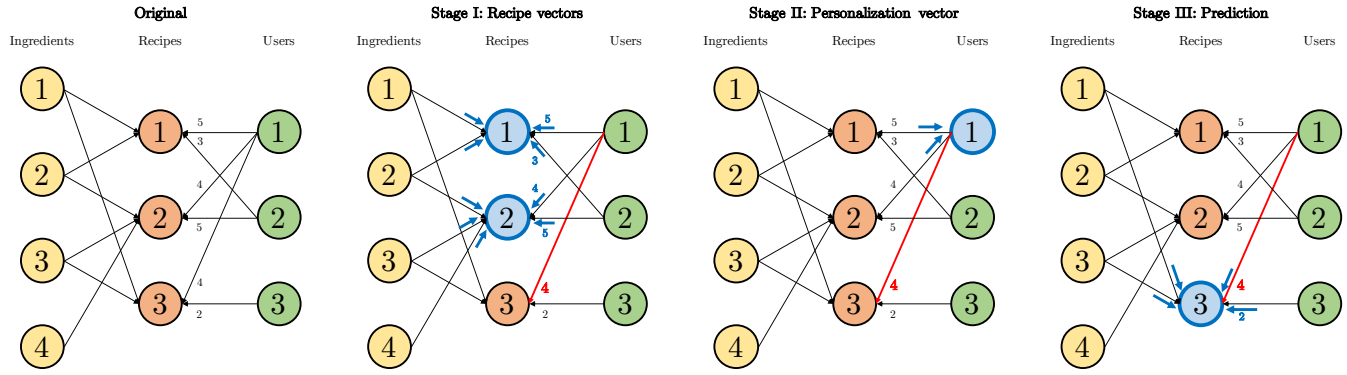

Figure 14: Training of GNN on ingredients, recipe and users data.

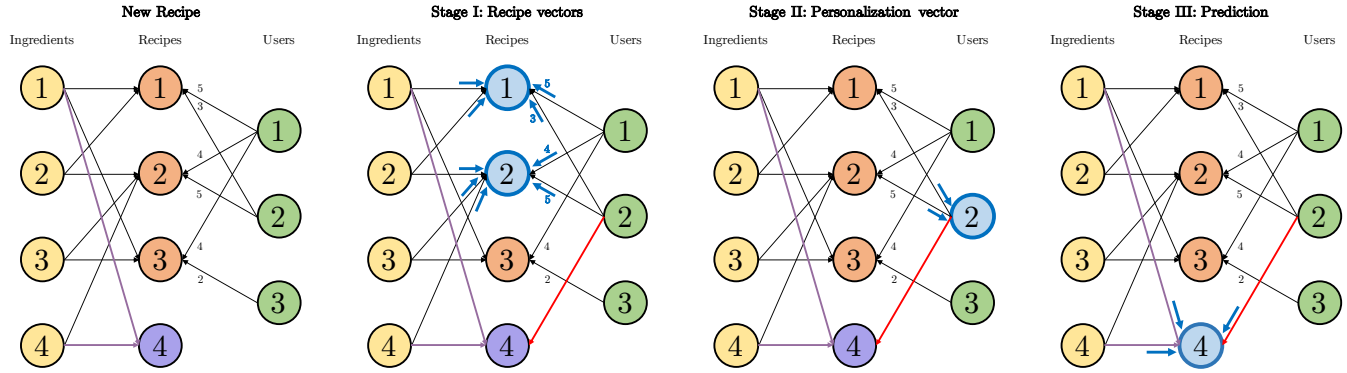

Figure 15: Inference on new recipes (purple node) which are not in the training data.

$$\overrightarrow{r_v} = \text{MLP}_1\left(\left[\text{AGG}(\{\overrightarrow{i_m}, \forall m \in \mathcal{N}_{IR}(v)\}); s_{u_*,v}; \text{Mean}(\{s_{m,v}, \forall m \in \mathcal{N}_{RU}(v)\})\right]\right), \tag{2}$$

where $\mathcal{N}_{IR}(v)$ and $\mathcal{N}_{RU}(v)$ are the neighbors of v in the ingredient-recipe and recipe-user graphs respectively, $\overrightarrow{i_v}$ is the shallow ingredient embedding for ingredient v that allows us to capture the semantics of the ingredient, AGG is the mean and max aggregation function for messages from neighbors, and MLP_k is the kth multi-layer perceptron that extracts a high dimensional feature representation for our inputs.

Stage II Personalization vector (3rd from left): In the previous stage, we have computed individual recipe vectors from the ingredient embeddings and it has some personalization element through the ratings score. We will now aggregate the information from each of these recipes that have been rated by the target user u_* to compute a personalization vector

$$\overrightarrow{u_*} = \text{MLP}_2\left(\text{AGG}(\{\overrightarrow{r_m}, \forall m \in \mathcal{N}_{RU}(u_*), m \neq r_*\})\right) \tag{3}$$

for the user. This vector represents the user's preferences and can be used to model the user's predictions on new recipes. Note that we can generate this personalization vector as soon as the user has given a rating on a single recipe in our database.

Stage III Prediction (4th from left): Finally, we combine the information about the recipe from aggregation of the ingredient embedding information, personalization vector of the target user $\overrightarrow{u_*}$ and rating information of other users by passing them through a MLP network which will predict

$$\hat{s}_{u_*,r_*} = \text{MLP}_3\left(\left[\text{AGG}(\{\overrightarrow{i_m}, \forall m \in \mathcal{N}_{IR}(r_*)\}); \overrightarrow{u_*}; \text{Mean}(\{s_{m,r_*}, \forall m \in \mathcal{N}_{RU}(r_*), m \neq u_*\})\right]\right), \tag{4}$$

the estimate of the rating of user u_* on r_*.

Training Loss: During training, we use CE, the cross entropy loss, on our predicted probability distribution $\hat{s}_{u_*,r_*}$ against the true distribution s_{u_*,r_*}.

$$\mathcal{L} = \text{CE}(s_{u_*,r_*}, \hat{s}_{u_*,r_*}) \tag{5}$$

Inference on novel recipes (Figure 15): The updated graph on the left shows what happens when there is a new recipe (purple node), note that there are no ratings from users

into the new recipe. The figure shows that we can perform the same three stage computation to predict user 2's rating on the new recipe (red line). Even when there are new recipes or new users, we can build the computational graph on the fly and perform inference with a single forward pass.

Recipe Recommendation Using Nutrition and Ratings Models

After we have our nutrition model and ratings model, we can make use of these two models to generate healthy and tasty recipes. To achieve this goal, for any user U and given any recipe R, we can define an objective function

$$G(R) = H(R) + c \cdot \text{Rating}(R, U).$$

Here, $\text{Rating}(R, U)$ is the rating U gives to R, and $c > 0$ is a constant picked by user depending on user's preference. $H(R)$ is the nutrition value we intend to optimize for (some value that satisfies "the higher the better", e.g. how much protein the recipe provides). This nutrition value can be a combination of different nutrition values. For instance, if one wants to have a low sugar and low sodium diet, then one could set $H(R) = -\text{Sodium}(R) - \text{Sugar}(R)$, where $\text{Sodium}(R)$ and $\text{Sugar}(R)$ are the sodium and sugar content of R, respectively. For our needs, because we are mainly looking at four types of nutrition values (fat, protein, sodium, sugar), we set

$$\begin{aligned} H(R) = c_1 \cdot \text{Fat}(R) + c_2 \cdot \text{Protein}(R) \\ + c_3 \cdot \text{Sodium}(R) + c_4 \cdot \text{Sugar}(R), \end{aligned} \tag{6}$$

where c_1, c_2, c_3, c_4 are constants that a user can pick depending on its own needs. Typically, $c_1, c_3, c_4 \leq 0$ and $c_2 \geq 0$ because one wants to minimize fat, sodium and sugar content while maximizing protein content.

One way we can maximize this objective function is by greedily swapping ingredients already in the recipe with other ingredients for at most K times, so as to maximize this objective function. This greedy approach might not give a globally optimal recipe, but it is can be interpreted easily and is highly efficient in terms of run-time.

In this way, we can help create and recommend recipes that suit one's taste and health needs. In terms of evaluation, we mainly use qualitative metrics to evaluate our result in recipe recommendation because of the subjectivity of the goodness of recipes and because we are generating novel healthy recipes which are unlikely to be in the recipe database.

Results

In our evaluation, we test the prediction performance of our ratings model by computing error metrics such as mean absolute error and root mean squared error on the actual ratings of recipe-user pairs in the test set. We will be comparing our ratings model with the collaborative filtering baselines and we know that our ratings model is good when we achieve lower error rates. While we are interested in evaluating the quality of recipes generated against baseline recipe substitution methods in (Elsweiler, Trattner, and Harvey 2017), it is difficult to do so because of the subjectivity of the goodness of recipes, and because we are generating novel healthy recipes that are unlikely to be in the recipe database. Accordingly, we have chosen to focus more on qualitative results from our method.

Nutrition Model

As discussed above, we built a linear model in order to infer the nutritional content of ingredients which is not available in the dataset. The following word clouds in Figures 12 and 13 are visualizations of the inferred nutritional contents for different ingredients using our linear model. For each ingredient in the word cloud of a nutrient, the size of the ingredient is proportional to the amount of the nutrient for that ingredient. We observe that pastry, rice and bun are in the carbohydrates word cloud and different steaks and chickens dominating the protein word cloud, which is generally in line with what we expect.

Ratings Model

Algorithm vs Metric	MAE	RMSE
KNN CF (Memory-based)	0.614	0.885
SVD CF (Model-based)	0.617	0.819
GraphSAGE on bipartite graphs	0.463	0.934

Table 2: Ratings prediction performance on test set.

Ratings prediction is a popular task that is commonly tackled by collaborative filtering algorithms. We implemented two collaborative filtering algorithms to predict ratings of users on unseen recipes given how they have rated other recipes in the past. The two baselines are: a memory-based method using K-Nearest Neighbours (KNN) and a model-based method using Singular Value Decomposition (SVD). In Table 2, we show the results on two metrics, Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). It can be observed that the performance of both methods are fairly even on MAE but SVD CF outperforms KNN CF slightly with lower error on RMSE by 0.819 to 0.885. Our GraphSAGE-based method outperforms the collaborative baselines on MAE by a decent margin of around 0.150 but performs slightly worse on RMSE, suggesting that our algorithm might have substantial error for a minority of the examples, but it produces less error for majority of the examples.

Recipe Recommendation using Nutrition and Ratings Models

To recommend healthy and tasty recipes using our nutrition model and ratings models, as stated previously, we try to maximize

$$G(R) = H(R) + c \cdot \text{Rating}(R, U),$$

where

$$\begin{aligned} H(R) = c_1 \cdot \text{Fat}(R) + c_2 \cdot \text{Protein}(R) \\ + c_3 \cdot \text{Sodium}(R) + c_4 \cdot \text{Sugar}(R), \end{aligned} \tag{7}$$

In practice, because using the ratings model to generate ratings $\text{Rating}(R, U)$ takes a considerable amount of time,

Yummy lasagna	Fat	Protein	Sodium	Sugar	Rating	Scoring
lasagna noodle, ground beef, ricotta cheese, romano cheese, egg, basil, garlic, sausage, mozzarella cheese	32.79	35.79	1050.60	4.59	4.02	397.49
lasagna noodle, bone-in ham, ricotta cheese, romano cheese, egg, basil, garlic, sausage, mozzarella cheese	51.83	64.55	2808.69	4.65	4.45	440.01
lasagna noodle, bone-in ham, ricotta cheese, romano cheese, egg, basil, garlic, sausage, bottle zesty dressing	57.07	61.10	2909.74	3.26	4.54	450.34

Figure 16: Recipe recommendation for user number 268713 starting from *Yummy Lasagna*.

Ingredients for Banana Maple Nut Bread	Fat	Protein	Sodium	Sugar	Rating	Scoring
Wheat flour, salt, oil, pure maple syrup, egg, milk, banana, walnut	14.49	8.09	234.60	23.48	4.27	177.72
Wheat flour, oat bran, oil, pure maple syrup, egg, milk, banana, walnut	14.49	8.59	42.84	23.48	4.42	384.54
Wheat flour, oat bran, oil, pure maple syrup, egg, nectarine, banana, walnut	13.52	11.45	9.75	19.59	4.43	419.24

Figure 17: Recipe recommendation for user number 2043209 starting from *Banana Maple Nut Bread*.

Lentil stuffed potatoes	Fat	Protein	Sodium	Sugar	Rating	Scoring
rice, lentil, boiling water, butter, onion, mint, salt, ground pepper, medium tomato, oil, garlic	13.14	11.98	320.17	6.13	4.34	432.56
rice, lentil, boiling water, bone-in ham, onion, mint, salt, ground pepper, medium tomato, oil, garlic	37.67	52.59	2189.34	6.10	4.70	484.75
rice, lentil, boiling water, bone-in ham, onion, mint, salt, ground pepper, medium tomato, venison stew meat, garlic	34.08	82.13	2464.17	10.92	4.70	518.25

Figure 18: Recipe recommendation for user number 3760041 starting from *Lentil Stuffed Tomatoes*.

in order to perform as many experiments as possible in a limited amount of time, we use a little heuristics in the selection of potential ingredients for swapping. First, we filtered out ingredients that appear at most five times in the ingredients vocabulary we built in preprocessing. This makes sense because in recommending recipes, we want to recommend recipes consisting of common ingredients so that users do not struggle to find rare ingredients recommended. Second, we only looked at ingredients that are particularly favorable in some nutrition value. Specifically, we only used the 50 highest protein, 50 lowest fat, 50 lowest sodium, as well as 50 lowest sugar ingredients. This heuristic makes sense because we usually want to recommend recipes with healthy ingredients.

We performed experiments under a variety of objectives, considering nutrition values including fat, protein, sodium, sugar and fiber. Due to page limit, here we mainly present results under these three objectives:

1. Lowering sugar while enhancing rating.
2. Lowering sodium and fat while enhancing rating.
3. Lowering fat and increasing protein while enhancing rating.

For each of the objectives, we try to conduct recipe recommendation for two types of users: high-degree users and random users. Presumably, the GNN should be predicting the ratings better for high-degree users because there are more information available, and thus the recommendation might make more sense for high-degree users. For both types of users, we start from recipes the user has rated and do the swaps for $K = 2$ times to generate recommendations.

For the first lowering sugar objective, we set $c_1 = c_2 = c_3 = 0$, $c_4 = -1$ and $c = 100$. Notice that the ratio between c_4 and c is quite hard to determine, and this ratio is highly dependent on how much we value low sugar as compared with rating.

This objective is especially significant for sweet recipes, because sweet recipes contain more sugar. For savory recipes that contain less sugar, it makes sense that in our algorithm, the importance of rating might overshadow the importance of sugar because the sugar content is already low. This is the case with the swapping transformation for the recipe *Yummy Lasagna* rated by a high-degree user number 268713, as is shown in Figure 16.

In the figure, light blue color marks old initial ingredients, and darker blue color marks the substitute ingredients. Red color marks contents that becomes worse compared to previously, and green color marks those that becomes better compared to previously. Notice that in the first iteration, ground beef is swapped with bone-in ham due to user's preference for bone-in ham (as shown in the sharp increase of rating), despite that the sugar content goes slightly up in

the first iteration. In general, this swap makes sense because bone-in ham and ground beef are comparable ingredients, while the user prefers the former over the latter.

Besides lowering sugar, lowering fat, lowering sodium and increasing proteins are the important things to do. Under these guidelines, we set up the second and third objectives. In the second lowering fat and sodium objective, we set $c_1 = -1, c_3 = -1, c_2 = c_4 = 0$ and $c = 100$. An example of this is done on the recipe *Banana Maple Nut Bread* for another high-degree user number 2043209, as shown in Figure 17.

From this visualization, we can see that the two swaps conducted are salt to oat bran and milk to nectarine, respectively. Almost all nutrition values are changing in the favorable directions during the two swaps, and the resulting recipe aligns with human tastes and habits. This aligns with the fact that user number 2043209 is the user with the most ratings, and thus the GNN has a better embedding for it.

For the third objective, we set $c_3 = c_4 = 0$, $c_1 = -1$, $c_2 = 1$ and $c = 100$. In this objective, the naive choice picked by this algorithm is often replacing high-fat ingredients (e.g. butter, oil) with high-protein meat (e.g. ham, stew meat). This makes sense to some extent because this serves our nutrition goal while maintaining some savory tastes. However, in actual cooking, it might not be a practical choice to add meat instead of oil in recipes. Here is one of such examples: recipe *Lentil Stuffed Tomatoes* rated by a random user number 3760041, as shown in Figure 18. We can see that in both swaps fatty ingredients got swapped with high-protein meat. Innately, it is harder to do recipe recommendation for random users than for high-degree users because there is less information available.

In general, our recipe recommendation algorithm is performing in a reasonable way in coming up with healthy and high-rating recipes, although we are not sure about the actual practicality of the new recipes we generated.

Conclusion

In this project, we proposed a three-part algorithm to predict users' ratings on recipes and recommend healthier and tastier alternative recipes. We constructed a nutrition model that assigns various nutrition values to ingredients using constrained optimization of a linear model. We also designed and trained a GNN model to predict a user's rating score for a recipe based on an ingredient-recipe graph and a recipe rating graph we built from the recipe rating dataset. These two models were then combined to make novel recipe recommendations for users. With a greedy algorithm, we generated and visualized some reasonable and novel recipes for various users. With our algorithm, we hope that food recommendation can go beyond standard recipes within databases to innovative recipes which are better in both health and taste.

Contributions

Yew Siang: Report writing, Dataset preprocessing, Collaborative filtering baselines, Nutrition model, Formulation and implementation of GNN-based ratings model.
Anita: Report writing, inspecting data, generating data statistics and visualizations, implementation of recipe recommendation using nutrition and ratings models, poster.
Nicholas: Report writing, generating data statistics and visualizations, poster, debugging, brainstorming the problem space, cooking knowledge

References

Ahn, Y.-Y.; Ahnert, S. E.; Bagrow, J. P.; and Barabasi, A.-L. 2011. Flavor network and the principles of food pairing. *Bulletin of the American Physical Society* 56(1).

Elsweiler, D.; Trattner, C.; and Harvey, M. 2017. Exploiting food choice biases for healthier recipe recommendation. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval* 575–584.

Teng, C.-Y.; Lin, Y.-R.; and Adamic, L. A. 2012. Recipe recommendation using ingredient networks. *Proceedings of the 3rd Annuasl ACM Web Science Conference (WebSci'12)* 298–307.