COOKBETTER 2.0: Be your own chef!

Kapil Chopra
North Carolina State University
kchopra@ncsu.edu

Mohit Satarkar
North Carolina State University
mmsatar2@ncsu.edu

Rachit Agarwal
North Carolina State University
ragarwa7@ncsu.edu

Wan-Yi Yeh
North Carolina State University
wyeh2@ncsu.edu

ABSTRACT

Food is a primary need of all humans. People have always wanted good taste in their food along with the food keeping them healthy. Proper nutrition is essential for optimum nourishment and health of everybody. Lack of time, cooking knowledge, and health problems stop people from receiving this. Multiple ways are tried by people to remedy this issue, but most of these ways make the user compromise on either taste or health. Having proper tools for users to fulfill their food intake should be a given today. So in this paper, we have improved the system "CookBetter[3]" bot that helps user manage their diet by complete comprehension of their preferences. Along with a proper recommendation of food recipes in accordance to the user's taste, the system targets the improvement of the user's health. Also, users can have ideal recipes suggested to them based on the ingredients they have available.

Keywords

Slack, Cooking, Ingredients Network, Recommendations

1. INTRODUCTION

CookBetter[3] is a bot built on slack platform where people can look for recipes with specific ingredients. As people today have very busy schedules, and lead a very hectic lifestyle. This has led to them having increased health concerns. Additionally, time as a resource has become more constrained than ever. Relating the aforementioned factors with everyday meal selection and preparation poses quite a dilemma. With an increased number of people concentrating on balancing body weight, taking precaution towards heart related ailments and maintaining a healthy physique in general, it becomes essential for recipe recommendation systems to consider these preferences at an individual level. "Allrecipes" website with 1.5 billion visits per year and 95 recipe views per second and "Supercook" website with a recipe database of over half a million recipes indicate the popularity of recipe recommendation systems. However, these sites solely depend on the ingredient data that the user inputs ignoring the constraints of health and time.

The culinary domain is extensive and complex which increases the difficulty for the Recommendation Systems. The number of ingredients and their possible combinations coupled with the number of techniques to prepare them results in a considerably large amount of data to be handled. As a consequence, user's opinion on food items can vary quite significantly depending on whether they like savoury to sweet, if they have specific allergies, prefer protein to carbohydrates etc. We try to incorporate such preferences, along with a few others into our application. This keeps us akin to our main objective: thrive to provide a more personalized experience to the user.

Another important factor to consider is cultural differences in food habits faced by people. A huge number of food recipes come up once we consider these factors. Many ingredients are unique for the use of a specific culture. Some of these might not have been heard by other cultures at all. But people have always considered variety as the spice of their lives. So getting people involved into trying culinary delights from other cultures is always an attractive prospect. This leads to a challenge, of getting users working with ingredients of which they may not be aware.

1.1 Problem Statement

All the current recipe recommendations systems emphasize on the ingredients and their suitable combinations but with more people putting their focus on health, it becomes a mandate of sort for recommendations systems to consider their health and diet preferences. We have further improved the slack bot based CookBetter bot system by applying our own improved algorithm to suggest recipes more accurately based on the preferences of the users.

1.2 Proposed Solution

The work done by the previous team for the CookBetter bot was considering the health, ingredient and dietary preferences of the user, and suggest them specific recipes based on their response. We further improved the accuracy of the recipe suggested by providing a broader recommendation and the system to choose options from. We maximized this accuracy by applying our association data mining algorithm on the ingredients to optimize the recipe suggestions. Another important feature we improved is the usability of the system. The user will receive new suggestions based on their preferences from previous usage of the system, which will be learned by the system. Additionally, we would also get the reaction of the user about the experience they had with the recipe, so that we can optimize it for the future use of the recipe with the other users as well.

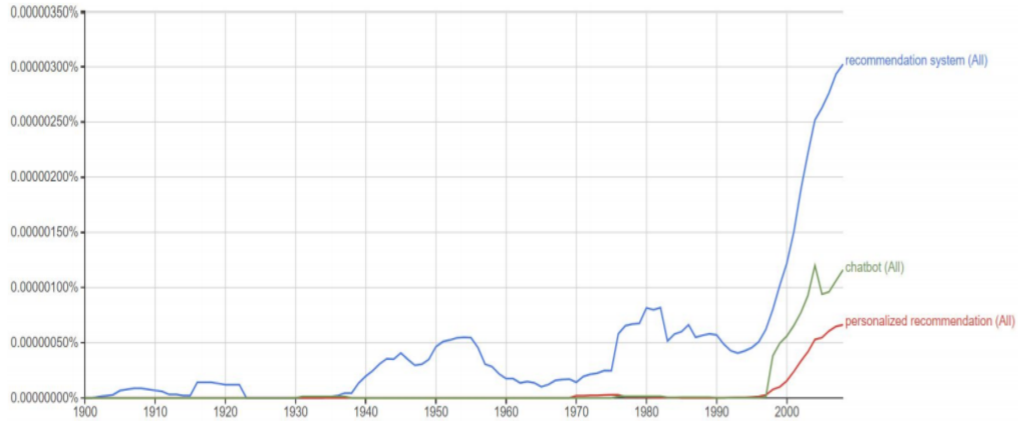

Figure 1: Results from Google Ngram Viewer.

2. LITERATURE SURVEY

2.1 Methodology

For the purpose of our literature study, we observed the work carried out by the previous team. Agreeing with their methods, we made use of tools: Google Scholar and Google NGram Viewer. We used Google Scholar for searching scholarly literature and finding papers relevant to our work. We made use of Google NGram Viewer for charting the frequencies of our search keywords mentioned in the vast amount of books available in the Google Books library.

2.2 Summary of papers

After perusing various research papers, the previous team discovered that a lot of research has been done in the area of recommendation systems, including recipe recommendation, and that there is a growing interest in this field. With the advent of machine learning, it has become possible for the system to learn the preferences of the user and improve the recommendation results. Although most of these papers proposed systems that solved one particular purpose, the methodologies adopted in each were significantly different. Although useful insights were gained from these papers, the decision was to try a different method to solve this problem - using chatbots. Plenty of work and research has been done in the sector of recipe recommendation on making the system more exclusive and unique to the user. The Paper by Mino et al.described a method considering the schedule of the user and calculating the intake or release of calories in each event which determined the next meal of the user[7], or the study by Yajima et al. on what a user will consider an easy recipe taking into account numbers of ingredients and seasoning in the recipe and its cooking time[10]. entails the results of their investigation by comparing three main recommendation strategies: content-based, collaborative, and hybrid[5].

In 2011, Ueda et al proposed a personalized recipe recommendation system based on the user's preference[1] and In 2012, Kuo et al investigated a menu planning mechanism based on user-speciïňĄed ingredients[9]. This experiment obtained positive results for the effectiveness of a recommendation system based on ingredients.

In 2015, Tome Eftimov [4] presents the analytical results of the ingredients matching in bakery products. This paper collected recipes from a free recipes web site and found the association rules between the recipes ingredient which was achieved by applying an Apriori algorithm and various visualization techniques to represent the discovered association rules. The paper covers data extraction, data preprocessing, association rules and visualization of the results during this work.

2.3 Results of n-gram viewer

The previous team chose the following keywords for our study: recommendation system, personalized recommendation and chatbot. The frequencies of mentions of these keywords in books published between the years 1900 and 2018 are shown in Figure 1. This time frame was chosen as there was either negligible or no mentions of our keywords in books prior to 1900. It was observed that the concept of recommendation systems was written about as early as 1902, and has been on a steady increase since 1992. However, personalized recommendation has only been mentioned since 1992, and has the least number of mentions among all our keywords. This indicates the need for further research and works on this subject. Chatbots, although researched upon ever since the 1960s by MIT professor Joseph Weizenbaum, have been popular in books only since 1996. The work on chatbots increased rapidly since then, hitting a peak in 2003.

3. USER STUDY

This section includes the relevant questions of the user survey conducted by the team who worked previously on CookBetter bot.

3.1 Would you like recipe recommendations that are personalized for your dietary restrictions?

This question was asked to determine if people would like an important feature of our project - personalized recipes based on dietary restrictions.

In the figure 2 shows that 80% of the participants require recipe recommendations that take into account their dietary restrictions. We assume that the remaining 20% do not have any strict dietary restrictions or are not particular about

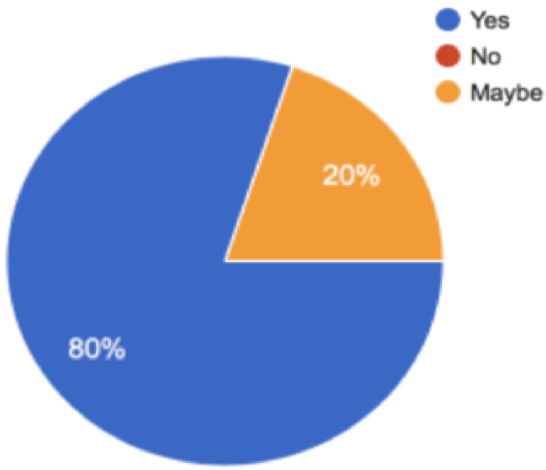

Figure 2: Would you like recipe recommendations that are personalized for your dietary restrictions?

this feature. However, majority of the participants have indicated their approval of a system that considers their health factors.

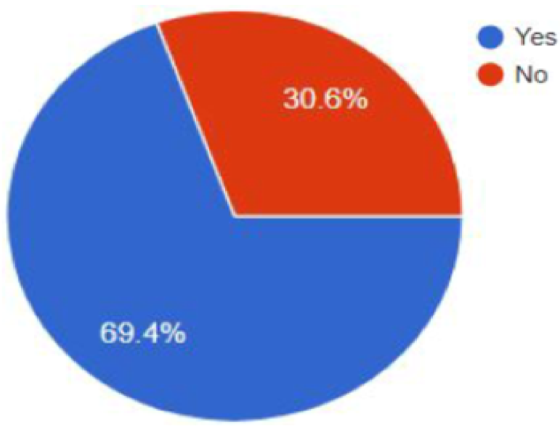

Figure 3: Do you find Slack user-friendly?

3.2 Do you find Slack user-friendly?

Since the chatbot proposed in this project is used through the Slack app, this question was asked to find out if the users found the application user-friendly and like using it, as the users of our chatbot are all going to be Slack users

It can be seen the figure 14 that 69.4% of the participants ïňĄnd the application Slack to be user-friendly. 30.6% of the participants have said that Slack is not user-friendly. This in- dictates the need to extend the chatbot to other messaging platforms which support chatbots in the future.

3.3 Would you prefer using a chatbot over a conventional website?

This question was asked to find out if users will adapt to our new proposed system of using a chatbot over the existing

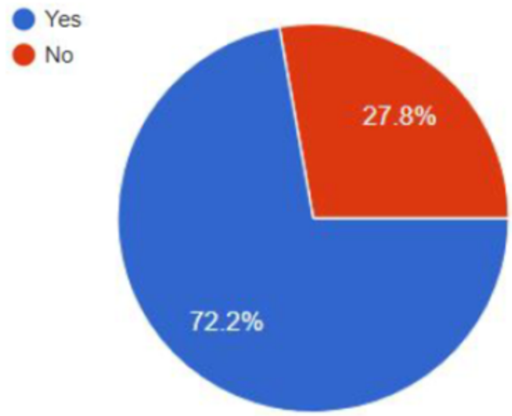

Figure 4: Would you prefer using a chatbot over a conventional website?

systems that use websites.

In the figure 4 depicts the percentage of users who would prefer using a chatbot over a conventional website that is available in most existing systems. 72.2% of the participants responded positively to this question.

Since, the results obtained for the above three questions asked by the previous group show that the users are interested in personalized recipe recommendations, slack platform and, a chatbot over a website, we decided to keep the architecture of the system unchanged.

4. MOTIVATION

Since the survey findings are inline with the current system architecture, we decided to work on the core functionality of the application i.e. the recommendation system. The entire project comprises of the recommendation system and the data it uses, so our plan was to validate the data and improve the recommendation system. The system was improved by using various data mining techniques to broaden the recipe recommendation range while improving the accuracy.

5. PROJECT ARCHITECTURE

In this section we have shown the architecture of the project. The model is shown in figure 5. In figure 5 we can see that the project uses Slack for creating application bots. Users utilize slack and chat with slack bot to interact with the application. Users use slash commands to enter into recipe recommendation function. At this point users will have to put in the filters he/she intends to use for the desired recipe. The bot then puts the desired request to the database. When the request is processed the user will receive step by step instructions on how to cook the recipe.

The database connected to slack bot comprises of integration of a recipe database with the application which sorts out all the filters for the recipe recommendation in database for rendering the results to get personalized recipe recommendation for the user. The application uses MySQL and Amazon Web Services for all the database applications purposes. User initially needs help to fill up a form which will create the basis of recommendation module will set up. User

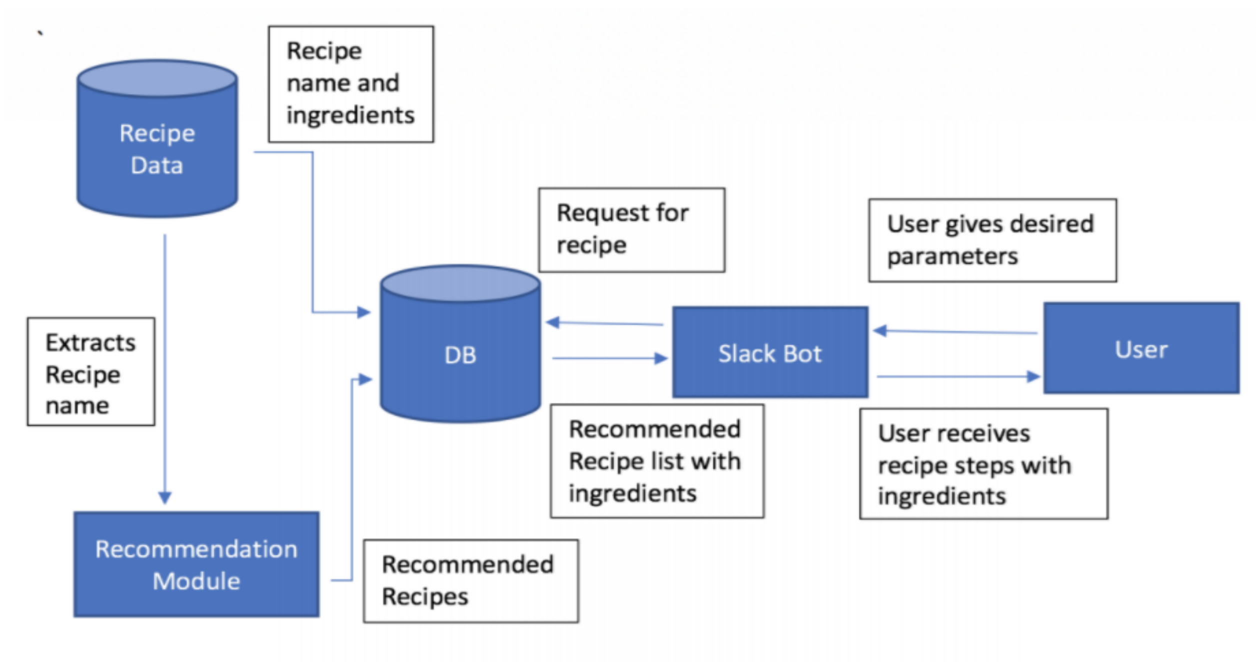

Figure 5: Project Architecture.

also has the option to negate these presetting and search for a recipe he desires by just typing its name. User can also search for recipes based on parameters like Time Constraint, Occasion or ingredients presently available to the user.

The two factor recommendation setup takes in global parameters Health, Allergies, Restrictions and Goal. These parameters are one time fill ups and are applied by default while user searches for a recipe. Here, health is if the user is healthy or suffers a disease like diabetes or not. Restrictions include dietary habits like vegan, non-vegetarian, etc. Goal indicates what user wants to achieve with the diet like reduce weight.

The second layer is the session filter which includes local parameters like time taken to cook, occasion, type of food, ingredients, etc. Here, occasion includes suggestion recipe for Halloween, Diwali, etc. Type of food corresponds to cuisine of choice of user. Ingredients indicate the list of available ingredients by the user.

All these parameters together constitute in fetching a recommended list of recipe for the user. The application is hosted on Amazon AWS. A cloud server has been used to provide account reliability and security. AWS has been chosen over other cloud services because of the features offered like auto scaling, elastic load balancing.

6. IMPROVEMENTS

6.1 /beyourownchef : Ingredients Network

We introduced this new feature to help user to experiment with the available ingredients at home. This feature will help user to know the best combination of ingredients that can be used to make a new recipe. User need to provide the list of ingredients and our model will analyze the input and provide the best combination of ingredients with the predicted rating to user. The entire process goes through five steps as shown in the figure. In this section we will discuss in detail about the process of data collection, reprocessing and implementation.

6.1.1 DATASET FOR INGREDIENT NETWORK

Epicurious.com is one of the most popular recipe sharing websites, where users can upload, review and rate cooking recipes. We downloaded 20000+ recipes of Epicurious from Kaggle.com. Each recipe of the downloaded data-set consist of the 11 attributes i.e. directions to cook, amount of fat, categories to which a recipe belongs, amount of calories, description, protein, rating, title, list of ingredients and amount of sodium. In order to understand the relationship between the ingredients we used the list of ingredient and a recipe rating to find the best possible combination and predicted rating.

6.1.2 DATA REPROCESSING

The first step in processing the available data-set is to filter out the list of ingredients of each recipe and making a count set of each ingredients to build the ingredient network. The second step in processing the list of ingredients is to build a list of relevant ingredient from the comma separated text provided by the user. To achieve this we initially calculated the frequency of each comma separated text in the list of 6000 recipes and removing all the ingredient with count zero. From this step we were initially able to remove the irrelevant text in the input.

Final step was to remove the list of ingredients which do not show particular complementary with any single group of ingredients. This step was important because these ingredients are so common that although they have many edge in the ingredient network but they are all weak. Few examples of such ingredients are salt and pepper. In this case the best accuracy turns out for the ingredients whose count is greater

/beyourownchef Chicken, thyme, butter, pepper, egg, bread, honey, salt, abcde

1. No ingredient found - abcde
2. Our algorithm doesn't consider pepper, and salt in analysis as these are used in most of the recipes.
3. According to the data, people do not usually prefer following ingredients together: honey with bread, egg, thyme, Chicken.
4. Optimum compatible ingredient found by algorithms are: butter, bread, Chicken, and thyme.
5. Predicted rating of the recipe cooked using the combination of suggested ingredients is 3.75/5

Figure 6: /beyourchef: Sample Run

than 62% of the recipes in the data-set.

6.1.3 MODEL IMPLEMENTATION

To build the ingredient network we used pointwise mutual information (PMI) [8] defined on pairs of ingredients.

$$PMI(a;b) = log(\frac{p(a,b)}{p(a)p(b)}),$$

where

$$p(a,b) = \frac{\#\,of\,ingredients\,containing\,a\,and\,b}{\#\,of\,recipes},$$

$$p(a) = \frac{\#\,of\,ingredients\,containing\,a}{\#\,of\,recipes},$$

$$p(b) = \frac{\#\,of\,ingredients\,containing\,b}{\#\,of\,recipes}.$$

This PMI value can be used to get the probability of each ingredient occurring separately that against the probability that two ingredients occur together. We used support and confidence criteria of the association rule to build the knowledge base, where confidence is the ratio between the number of recipes that have true values for all ingredients in X and Y and the number of recipes that have true values for all ingredients in X and support of an association rule is the ratio of the number of recipes that have true values for all ingredients in X and Y and the number of recipes in our database.

For each possible pair of ingredients we calculated the PMI value and divided the pairs of ingredient into three possible sets i.e. low, moderate and high complement set. The intuition of dividing the pairs into three sets is to find the complementary ingredients that would yield higher ratings, while ingredients that are in the low complement set would lower the average rating.

To make this division we picked 20 different threshold values using 6000 recipes and executed the model with the thresholds on remaining 14000 recipes to find out the percentage of ingredients falling in the same set as it was in the training data-set. Threshold value selected for common ingredients is 0.33, for low complement set is -0.15017 and for best combination is 0.0923. Accuracy for the following threshold value resulted out to be 0.6212.

Next we build a graph using these three sets where weight of each edge defines the respective PMI values between the vertex. Weight of each edge in the low complement set was set to negative and positive for the edges in high complement set. Further to find out the best possible combination of ingredients we build a union set of ingredients in the best complement set and updated the weight of the resulting set by adding the weight of the ingredient in the low ingredient set. Finally to get the best possible combination of ingredients we remove the edges of the graph with negative weights and return the connected component in the graph with maximum total weight.

6.1.4 Validation

1. We used k-fold cross validation to validate the output of our model. In this we divided the original sample is randomly partitioned into 10 equal sized subsamples i.e. division 14000 recipes. In next step we used one sub-sample to validate data for testing the model, and the remaining 9 sub-samples to train our model. The cross-validation process is then repeated 10 times , with each of the 10 sub-samples used exactly once as the validation data.

2. In each run, we computed the percentage of pair ingredients falling in the low, moderate and high complement set with training and testing folds. This validation was used to find out the best threshold values for each division of set to increase the overall performance of the model.

3. To build K-fold cross validation in Java we used Cross-Validation class of Java-ML library.

4. Currently, the overall performance of the implemented model is 0.6212.

6.1.5 Limitations of the implemented model

As the model is accepting the list of comma separated ingredients from the user, the performance of the model can be significantly reduced for the below cases.

1. The model works best for the ingredients in singular form but not for plural forms. For example, if the user includes eggs in the list of ingredients but not egg then the performance of the model will not be as good as in case of singular.

2. If the user includes ingredients with spaces then the model will not be able to recognize if the input has one ingredient or two. In this case, the model will completely fail to consider the input as one ingredient. For example, if the user enters boiled eggs, the model will treat this as two separate ingredients rather than one.

3. Any spelling mistakes in the input will be not considered as an ingredient in the model and so it will not be processed by the model.

4. Currently, the recipes of all cuisines have not been used in our data-set so our model may present the recipes of a certain set of ingredients.

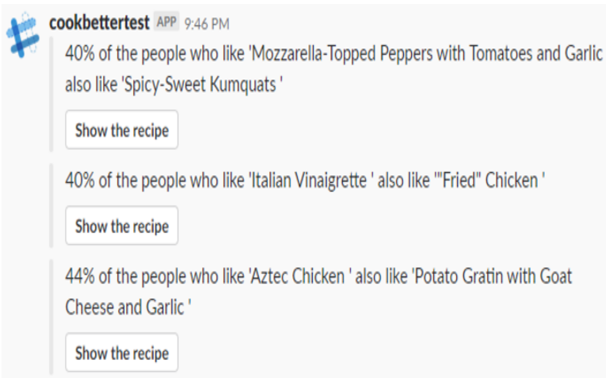

Figure 7: Recommendation snapshot

6.2 Slack command : /recommend

The recommend command operates on the list of recipes liked by the user. It uses that list to process the data-set in order to find other users who are similar to the user who needs recommendation in terms of a particular recipe in consideration. Ones a similar used is found, the algorithm starts recording the recipes that have been liked by the similar users. It iterates through the entire list of users in order to find similar users and recording the recipes liked by them. After an iterations, the algorithm has all the data it requires to make a recommendation based on the recipes like by the similar users. The recipe that has the highest frequency in the list of liked recipes of all the similar users is the one that is recommended and stored until all the other recipes, liked by the user who requested recommendations, have been processed. At the end, the algorithm retrieves all the recipes that it have stored and displays them to the user along with the initial recipe using which similar users were found and the percentage of the users who like the recommended recipe from the set of users who liked the recipe on which algorithm was initiated.

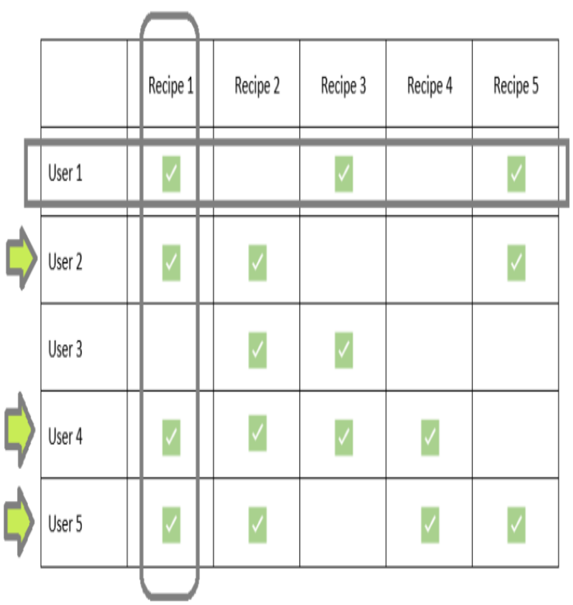

Figure 8: Dataset representation.

To understand the functioning of this algorithm, let us consider an example. The image above shows a 2-Dimensional matrix showing the list of users and recipes with a check mark at the intersection of a user and a recipe to indicate that this user likes the particular recipe.

This algorithm functions on two inputs: the list of recipes liked by the user who requires recommendation, and the entire data-set which is represented by the 2-Dimensional matrix shown above.

Let us assume that User 1 from the above mentioned data-set requested a recommendation on recipe. The first operation that is performed by the system is to extract the list of recipes liked by User 1 which is represented by the row 1 of the data-set. This list along with the entire data set is sent as the input to the algorithm.

Since there are three recipes that have been liked by User 1, this algorithms is run three times to find similar users and a recipe which is most popular among the other users who like each of the three recipes.

The first run of the algorithm is concerned with Recipe 1 which is a recipe liked by User 1. The algorithm finds the similar users by iterating through the entire list of users who have liked Recipe 1 which means [User 2, User 4, User 5]. First the algorithm processes the data of User 2. It starts recording the recipes liked by this user along with a count of 1 as each of the liked recipe has only appeared once. At the end of processing User 2, the data stored comes out to be (Recipe 2, 1),(Recipe 5). Then the algorithm proceeds to the next user in the list i.e. User 4 and does the same things. At the end of processing the data of User 4, the data stored comes out to be (Recipe 2, 2), (Recipe 3, 1), (Recipe 4, 1), (Recipe 5, 1). Similarly the user processes the last user in the list i.e. User 5. At the end of processing the data of User 5, the data stored becomes (Recipe 2, 3), (Recipe 3, 1), (Recipe 4, 2), (Recipe 5, 2). At this point, the algorithm has collected all the data it requires about the similar users. Now, it selects the recipe with the highest frequency which is Recipe 2 and computes the percentage of the similar users who have liked this recipe. Out of the four users, including User 1, three of them have liked this recipe so the percentage become 75. This recipe i.e. Recipe 2 is stored to the solution set along with its percentage i.e. 75 and the recipe for which it has to be recommended i.e. Recipe 1. So when this stored data is retrieved, it is displayed in the format that can be understood by the user very easily. The text that gets displayed for this data is ”75% people who like Recipe 1 also like Recipe 2”.

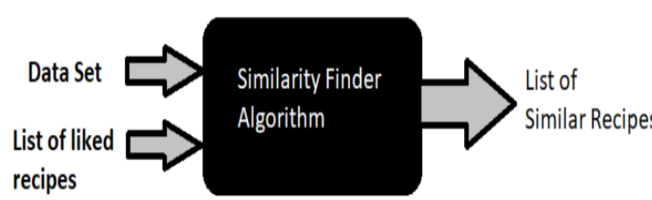

Figure 9: I/O representation

Similarly the algorithm performs the iterations for the remaining recipes in the list of liked recipes of User 1 which are Recipe 3 and Recipe 5, and obtains their similar users which are used to compute the percentage and select the recipe with the maximum frequency. All the data is then stored in the solution set in the manner similar to the Recipe 1.

At the end of the processing, the algorithm starts displaying the results. The results are produced for each of the recipe in the list of the liked recipe by the User 1. The final results obtained will have 3 recommendations for the user which have been personalized based on each of the recipe that they have liked. The final result for the current example that we have considered is displayed below. "75% people who like Recipe 1 also like Recipe 2". "66.7% people who like Recipe 3 also like Recipe 2". "66.7% people who like Recipe 5 also like Recipe 2".

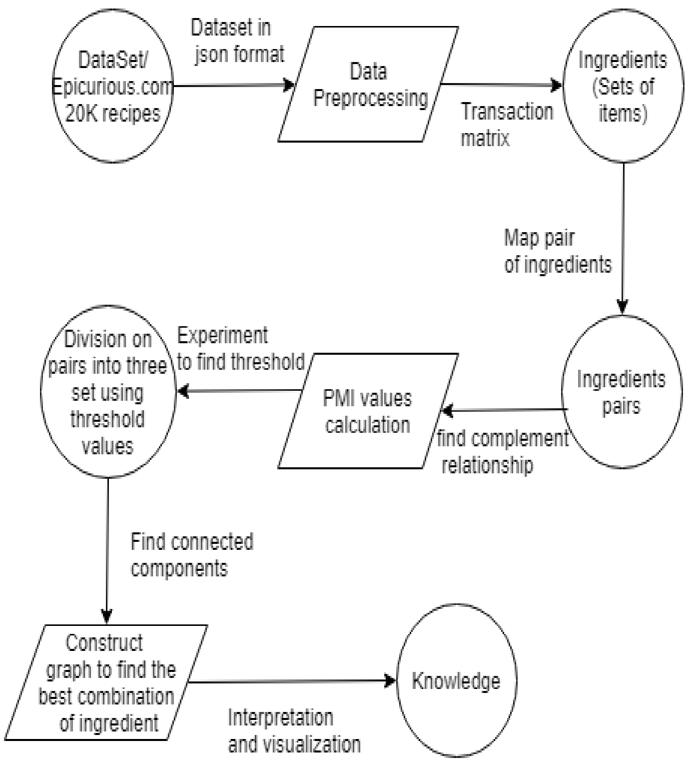

Figure 10: The knowledge discovery process.

6.3 Usability

6.3.1 Chat interface rather than filters.

The chatbot previously used drop down menus and buttons as the input medium for the required information. While these drop down menus can be helpful in maintaining a clean interface, it increases the user effort required to provide some key words. A better technique for input in this case would be an input from a text box. While chatting with the bot, the users can enter the list of ingredients separated by comma whenever the bot asks them to. It saves a lot of effort of opening the drop down menu and selecting an option.

We implemented a new slack command /beyourownchef to provide the user an interface to enter list of ingredients in a comma separated plain text. This input string is parsed by our parsing algorithm and a list of required data, such as the list of ingredients, were obtained and recommended the best list of ingredients and the relevant recipe. After this change, we improved the user experience by replacing the drop-down boxes with a plain text area.

6.3.2 Updated User Interface

The users could have the best experience using the application if they received all the ingredient and recipe details directly onto the slack interface being utilized for the application. While the previous version seemed effective when the final result was a link to the recipe instructions on the website Epicurious, we felt providing the description of recipe, all ingredients, rating, calories, ingredient specifics, and ailment and allergy specifics on our own interface would make the user experience more efficient and increase the usability of the application(see figure). We also kept the website link to access the completed recipe.

6.3.3 Alternate search result

The other improvement is the alternative search result. When people search a recipe by ingredients, they tend to input the ingredients which they had at that time. If thereâĂŹs no recipe contains all the ingredients, they have to search recipe by different combinations which would be a bored and tedious work. Our system has the ability to exclude ingredients and show the alternative result for the user. For example, if the user canâĂŹt find the recipe include both beef and asparagus. Instead of showing a message of not finding the result, our application can exclude beef and search the recipes by asparagus.

6.3.4 Short description

Whenever a user uses the bot for recipe recommendation, they are provided with links to the recipe page on a website. The user is then required to visit that website to see the recipe. This becomes a tedious task which leads to bad user experience.

SO we changed this by providing a short description of the recipe to the user. It will help the user to know the ingredients and the time required to make that particular dish. This information will save a lot of time for the user as it does not involve visiting a website for each of the recommended recipe.

7. DATA SETS

The datasets used to populate the database for this application has been taken from the website "Epicurious" and "Kaggle" [2] which provides data about recipes, their rating, nutritional content, and categories. This data consist of 20k recipes listed by recipe rating, nutritional information and assigned category (sparse). The database also contains the UserID which is obtained from Slack along with the ailments, allergies, weight goals, dietary restrictions.

8. PLAN

We followed Agile methodology of development to deliver 9 user stories in 15 days which includes all phases like development, testing, validation and deployment. The project is managed on GitHub, with issues created and assigned to team members based on the specific part of the project they had experience working in. During planning phase, we felt that a few principles of the spiral model would help our purpose a lot. Specifically, commit partitions helped in proper analysis of the new functionalities due to which the system

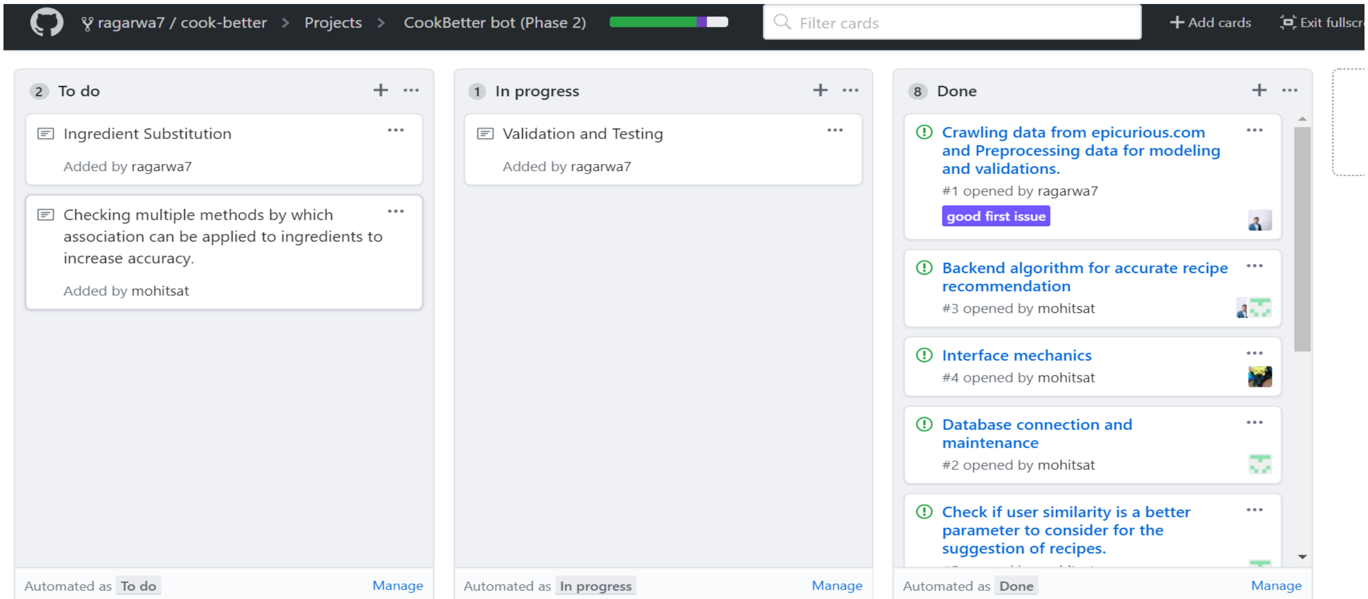

Figure 11: Story board.

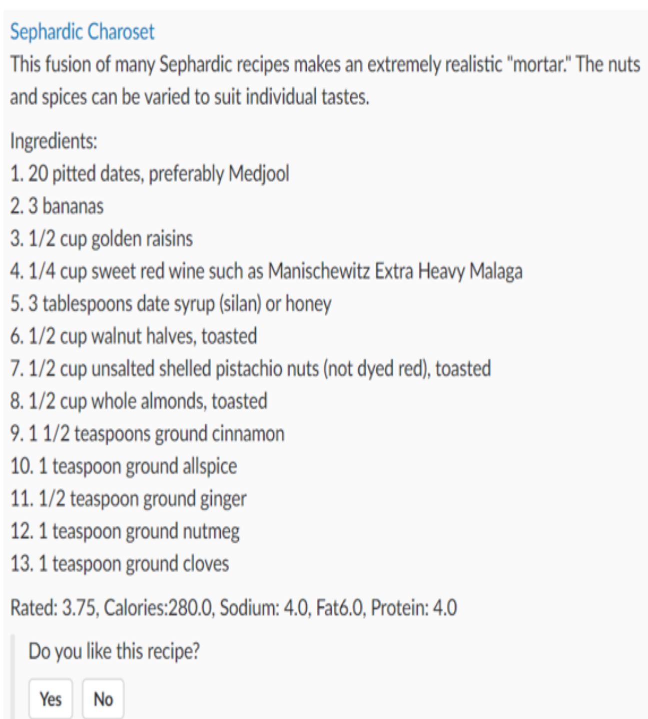

Figure 12: New User Interface

would considered beneficial. Once the new functionalities are finalized, we will be working on different stories built to improve the functionalities of the existing system.

The designed 9 user stories which include all the functionalities (enhancement and new features) of the application were divided in 2 sprints of 7 days with each sprint targeted to deliver stories worth 20 story points. We used Planning Poker estimation technique to find out the number of stories we can deliver in the given time-line. Finally, each developer will be assigned 2 stories at the starting of each sprint and developer will be responsible for moving each story to relevant phases on the story-board (Figure 11). To make sure that each developer can progress with the story and to tackle any blockers as soon as possible, we have planned scrum meeting every alternate day.

Moreover, we followed pair programming technique for implementation of stories related to data mining as these stories cannot be build in parallel as we need to prepare data before implementing model and so on. So we plan to include pair programming technique of Agile methodology which helped us to raise the communication bandwidth and frequency within the project, increasing overall information flow within the team.

9. EVALUATION

Any analysis of the application for measuring success or effectiveness rested on a diligent and thorough evaluation of the application from multiple different types of users. Reactions of intended user base would always receive more focus from the developers though, so these evaluations help out a bit more. A proper application of the gathered results would result in a much more used and robust application, satisfying the intended user base, and also comprehending the steps required to develop the app for other user bases too.

Hence, we extensively studied User Experience principles so that we could properly apply them in our evaluation process. We recruit a few participants for interview, and others for questionnaire only. The advantage is that by interview people, we can directly observe their reaction when them using our application. From the research of usability testing[6], we learn the application of some principles relevant to our evaluation process, which included:

1. **Optimum number of evaluators** For the first phase

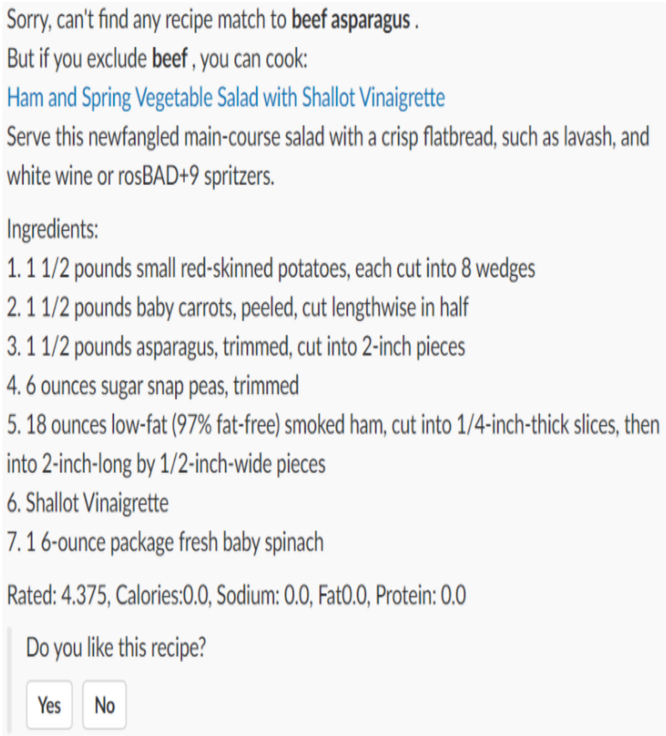

Figure 13: ALternate Search

of evaluation, we need only five participants to evaluate our product extensively. According to the research, we discover more than 85 percent of the core usability problems using the 5 relevant evaluators. It's an efficient and effective way to evaluate our application and improve it based on our findings.

Once we finished these, we set up the final evaluation environment. We presented the results of this in the class presentation. We improved the accuracy of our application from less than 52.1% to 62.12% based on our findings from the first evaluation. We also found out that 89% actually expected the recipe suggested by the system.

2. **Prevent evaluator bias** Preventing evaluator bias is a crucial measure which is required to prevent incorrect evaluation of your product. The first thing we did to complete this was that we ensured the evaluators were open and comfortable in their responses during the evaluation process. We didn't set any time limitation and made sure the participants understood that it's an evaluation of the application, not an evaluation of the evaluators themselves. Gamifying the evaluation process was something which also help us make the evaluator comfortable.

   One important measure was that no more than two people amongst us, i.e., the developers, would stay with the evaluators in the room when evaluations were being conducted. This made the atmosphere more conducive for the user. If the number of developers is larger than the number of evaluators, the evaluator feels the need to not trust their instincts about the application evaluation. They feel the need to agree with the evaluators instead of being completely honest about their responses.

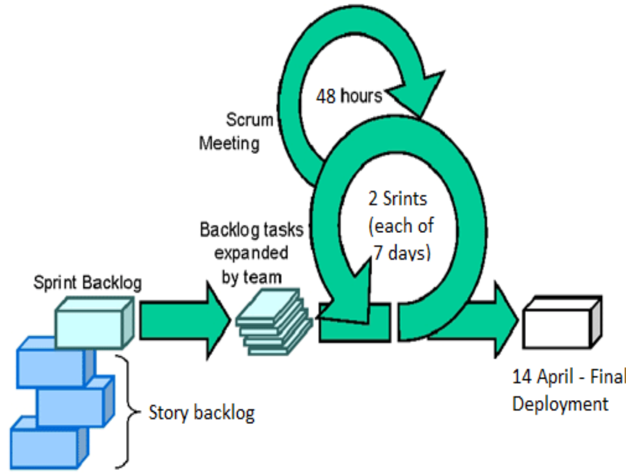

Figure 14: Plan: Agile

3. **Evaluation environment** In the interviews, We set up tasks for the evaluator to carry out, and recorded their responses for each of the assigned tasks. The tasks we assigned included test runs of our system to help us see if our algorithm is working as we have designed it to be. The purpose of the tasks was to test the ingredient input against the ingredient network we have designed to see if we are able to suggest optimum ingredients which are compatible to use for culinary recipes. We asked the user to complete all the tasks 5 times, to help us receive more data for being able to test our ingredient network algorithm. The tasks we assigned included:

   (a) Please paste the list of ingredients you entered. e.g. "Chicken, thyme, butter, pepper, egg, bread, honey, salt, abcde"

   (b) Please paste the report you received for the first input.

   (c) First two lines of the report give the list of inappropriate input and common ingredients. Please list out the ingredients you found which are not common or not an inappropriate input but suggested otherwise. (Type NA if none)

   (d) Fourth line of our report lists the best combination of ingredients which can be used in a recipe. Did you know any good recipe that uses these ingredients (Use google for help, if necessary)?

9.0.1 Helper Command - /recommendpopulate

For the /recommend command to work, it requires some data about the liked recipes of the user. But when a new user joins the slack workspace, they do not have any data about the recipes they like. This was a challenge we faced while preparing the system for testing.

To tackle this challenge two solutions we came up with two solutions:

1. Develop an iterative procedure for the evaluators to follow which includes testing the other commands and liking a few recipes in order to populate the database.

2. Displaying a small list of recipes for the users to checkout and like and use that data to populate the database.

One thing that was common in these two proposed solutions was that they required the recipes to have a like button which has a connection to the recipe ID and database. Since this feature was being added to improve the quality and ease of recipe recommendation for the user, it had to be added with the initial add-on being the like button.

The next step towards adding a feature was to test the usability of the system after incorporating the feature so that the user experience can be studied. In order to do so, we created paper 2 paper prototypes - one of them included the first proposed solution, and the other had the second solution. We then selected 5 people at random to evaluate our prototypes. This step was necessary to see the impact of these two features on the user experience. The feedback we received from the users wasnâĂŹt skewed towards any specific solution as some users understood that they needed to like some recipes to their list of liked recipes while the others quickly jumped to the /recommend command and were disappointed to see just 1 or 2 recommendations. Since all the users were not satisfied with the main functionality of recommendation of the system that we had created due to the their struggles with one feature, we decided to incorporate both the proposed solutions in our system in order to enable it to satisfy the usability types of a wider user set.

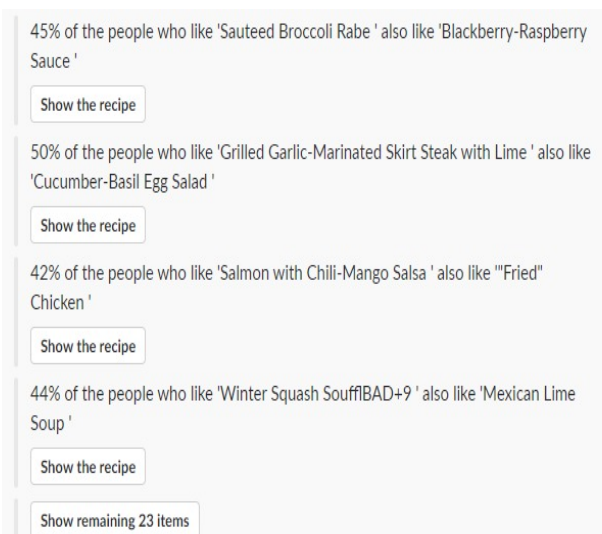

Figure 15: /recommendpopulate for evaluation

The first feature of following a procedure to fill the user data about the liked recipes was implemented by adding the like button next to each of the recipes in the /searchrecipes command as this was the first think that we had planned for the users to use. This provided the opportunity to add data to the list of liked recipes by liking any of the suggested recipes.

The second feature was implemented in the form of a new slash command in slack - /recommendpopulate. Its task was to provide the user with a list of 50 recipes and giving them the ability to like any of those recipes. This was observed to reduce a lot of work on userâĂŹs part that was required to populate the database. Now instead of going through a set procedure to populate the database the user had the ability to just enter a command and start liking the recipes.

Once these tasks were completed 5 times, we asked the evaluators to try the /recommend feature multiple times, and recorded if the suggested recipes were something that the evaluator liked as well.

9.1 Evaluation results

After the complete set of evaluations, we found out that our system was 62.12% accurate. This was close to the values which we expected to get from our system based on the time we had for working on the system. We can improve this by training against a bigger data set, analyzing collected reviews and adjusting our threshold values according to our observations of the system runs.

The second surprising result was that 30% of the users did not agree with the ingredients which we suggested to them based on their compatibility. This was a result of their personal biases for food items which we did not properly consider, while we just used the complement network to see the ingredients with a higher compatibility rating. This can be corrected by properly profiling all the user data.

A third important result we found was that 65% of the evaluators were able to think of a recipe based on the ingredients we suggested them after using our complement network on their ingredient input. This helped us understand that our algorithm was efficient for suggesting proper ingredients on which recipes could devised. This can be further improved to properly suggest recipes based on our ingredient suggestions directly.

10. FUTURE WORK

1. Improving accuracy of the predicted rating and recommendations from 62.12% to at least 70% by adding more sets of recipes and shifting the threshold values accordingly.

2. To increase the performance of the model, natural language processing could be applied to the list of ingredients entered by the user. This can further improve the user experience of the application.

3. Like button can be added to the result of the /beyourownchef command to collect the feedback from user which can be used to improve the performance of the mode.

4. Extend ingredient networks to incorporate the cooking methods as well.

5. Suggesting ingredient substitutes (by implementing ingredient substitute network) i.e. describe the recipe modifications extracted from user reviews, including adjustment, deletion and addition.

6. Share your own recipes. Using the ingredient network, user is able to figure out the best combination of ingredient that can be used to prepare a new recipe. If user find out that recipe is good enough to share with everyone, the user should have a platform to share a recipe.

7. Comment system. This can help users to improve the recipe by having comments from different users.

11. CONCLUSION

In the modern world, a delicious meal is the key to please everyone but to get the ingredients in the right amount while cooking is just a trial and error process. People have learned that this is an exhaustive method so it is wise to record the best results and share it with others so that it can save them a lot of time. Due to this wisdom, the internet has become a great platform to share the recipes. There are a lot of websites which provide a massive collection of recipes for the people to use. In this project, we have studied and processed this data to yield a system as the result, that is capable of providing the user with a recipe that is personalized according to the user's taste and health restrictions. This system has an intuitive user interface which has been created on slack and it has a fast and accurate implementation of data mining algorithms working on a gigantic data set of over 20000 recipes to yield an accurate result.

12. REFERENCES

[1] *https://books.google.com/ngrams*, `https://books.google.com/ngrams`.

[2] *Recipe data-set*, `https://www.kaggle.com/hugodarwood/epirecipes`.

[3] a. R. K. Aayush Patia and S. Vyas. Cookbetter: A bot for personalized recipe recommendation. *Software Engineering, NCSU*, Spring 2018.

[4] T. Eftimov and B. K. Seljak. Ingredients matching in bakery products. *18th International Multiconference on Information Society*, 2015.

[5] J. Freyne and S. Berkovsky. Intelligent food planning: personalized recipe recommendation. in 15th international conference on intelligent user interfaces. *"15th international conference on Intelligent user interfaces"*, October 2011.

[6] N. J, Landauer, and T. K. A mathematical model of the finding of usability problems. *"INTERACT'93 and CHI'93 conference on Human factors in computing systems, ACM"*, May 1993.

[7] Y. Mino and I. Kobayashi. Recipe recommendation for a diet considering a user's schedule and the balance of nourishment. *"IEEE International"*, December 2000.

[8] C.-Y. Teng, Y.-R. Lin, and L. A. Adamic. Recipe recommendation using ingredient networks. *"International conference"*, May 2012.

[9] M. Ueda and a. S. N. M Takahata. User's food preference extraction for personalized cooking recipe recommendation. *"Workshop of ISWC"*, pages 98–105, February 2010.

[10] A. Yajima and I. Kobayashi. Easy cooking recipe recommendation considering user's conditions. *IEEE/WIC/ACM International Joint Conferences*, October 2009.

12.1 Chits

1. SHX
2. LQX
3. SSM
4. AKO
5. TWH
6. JFA
7. IJE
8. TTY
9. ONH
10. UJT
11. ORV
12. ZKW
13. PWN
14. DCM
15. WBM
16. APM