Network Tour of Data Science : A Network Tour of Cooking

Couyoupetrou Julien, Mueller Gauthier, Renaud David

January 2020

Abstract

This document presents a network approach of cooking recipes, using the Recipe1M dataset [2]. After a presentation of the dataset, the graph creation is detailed. The graph is explored and is used to create recipes. The created recipes are then evaluated and compared to existing recipes.

1 Introduction

Cooking is "the art, technology, science and craft of preparing food for consumption"[3]. The cooking techniques and ingredients have evolved across the ages and is specific to each culture. The way to cook and to eat food is a very important aspect of many cultures, as the meals are an important moment of everyday life, and cooking is a very old human occupation. There are archaeological evidence of human cooking more than 1 million years ago [1], and the first ever cooked meal could have more than 2 million years. Today, with the new technologies, new tools and ways to cook are appearing : Electrical oven, microwave, freezer, molecular cooking and others are for example recent tools and techniques that have revolutionized cooking. In addition, with the globalization, we have access to more ingredients than ever before, and to the cooking culture of most of the countries of the world. We also have better and better knowledge of whether an ingredient is healthy or not, and thus the possibility to create new recipes, or to modify existing recipes using this knowledge. The major obstacle to be efficient in this approach is the quantity of existing data about recipes, health indicators, and cooking techniques: there are thousands and thousands of ingredients and recipes available freely on the web. Thus, the best way to approach efficiently this problem could be the approach of the data scientist instead of the one of the chef.
This report present a graph approach to cooking recipes using the recipes1M dataset [2] containing over 1 million recipes, and try to answer the following questions: Is it possible to use a network approach to create new recipes based on what is left in your fridge? To generate recipes from scratch? We will also explore the connection between the ingredients, which ingredients are most often used together and why, and the general properties of the recipes graph.

2 Presentation of the Recipe1M dataset

To deal with these questions, we downloaded the Recipe1M dataset [2], which contains circa 1 million recipes, with 13 million images. For a matter of storage on our personal computers, we will not use the images in this project. (The images represent 3.36 Tb of data). The original purpose of this dataset was to generate recipes from pictures or food. The opposite approach also has been tried, generating pictures based on recipes.

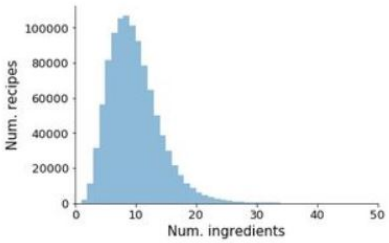

Figure 1: Number of ingredient distribution of the recipe [2]

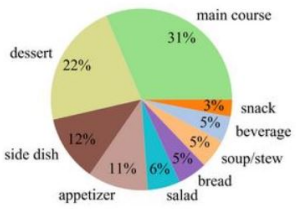

Figure 2: Types of recipes in the dataset [2]

We can see on figure 2 that the recipes in the dataset contain around 10 ingredients in average and up to 30 ingredients. The figure 2 show that the most represented type of

recipes is the main courses (31%), followed by desserts(22%), side dishes (12%) and appetizers (11%).

3 Cleaning of the dataset

In order to be able to work locally, both memory and time-wise, we had to reduce the size of the recipes dataset. This was accomplished by randomly sampling a subset of 10 000 recipes. Results with a higher number of recipes can be obtained easily by sampling a different number of recipes in the file "clean_data.py" on the Github repository. The dataset Recipe1M has been made by scraping many different cooking websites. The main advantage of this approach is the huge quantity of free data available online. The major drawback is the lack of structure and consistency in the recipes, resulting in a difficult cleaning, and a lot of outliers. Many recipes contain typos, the same ingredients are present under different small variations, for example, vanilla extract, vanilla essence. Moreover, some recipes are written in different languages, resulting in different representations for the same ingredients. Not only, but there are also recipes that consist of only one or two ingredients because the published recipe did not follow the same structure and the list of ingredients were directly written in the instruction field.

Besides, the ingredient field of the dataset contains some details either between parenthesis or some specific format, which differs from recipe to recipe, and also quantities, in many different units. Sometimes even the brand of the product is present, which we had to manually look for and remove. Thus, the dataset is highly noisy. In order to clean it without having to manually invest too much time, we used a Natural language processing library called Spacy and regular expressions. We started by removing indication between parenthesis. Afterwards we try to remove the aforementioned units and annotation by keeping only words that were annotated by the NLP library as noun, removing digits, dots, adjectives, verbs and words terminating with "-ing" and "less". Still, many nouns had to be manually removed by adding them to a stop words list.

Hence, we are able to transform sentences similar to "Two spoons of sugar" into "sugar", effectively reducing the inconsistencies. Furthermore this approach allows us to keep bi-grams, like "bell peppers", "almond butter", and some trigram, like "yukon gold potato". However, many outliers remain. To take care of the remaining outliers we deleted all the ingredients that appears only once in the dataset, as they are probably not well cleaned ingredients. After this step, we had only 2754 ingredients left instead of the original 9663 ingredients. Thus, 71% of the sampled dataset is not well cleaned and not usable. This is caused by the previously discussed observations regarding the scraped data, which leads simple ingredients, like salt, pepper, butter, to be more likely to be scraped into a consistent ingredient name than more complicated ones, like chicken, that can be presented as chicken wings, chicken legs, fried chicken, chicken breast, graham chicken, etc..., thus, simple ingredients are likely to be over-represented in our graph, and more complicated ones to be considered as outliers since their names are rarely consistent. However, given the limited amount of time, we decided to work with these 2754 ingredients to have a first insight of the characteristics of the dataset. For future application, it would be possible to spend more time looking for outliers manually, and add custom NLP rules.

4 Creation/exploration of the graph

4.1 Analysis of the different steps

To create a graph based on the cleaned data-set, we have decided to connect together ingredients that appear in a sufficient number of recipes together. We start with the cleaned data-set, a matrix of (10000,2754), corresponding to 10000 recipes and 2754 ingredients, one recipe per row, one ingredient per column. Then, using this basis, we created a weighted adjacency matrix (2754,2754) counting the number of times the ingredients are appearing together in a recipe. First, we decided to observe the number of times each ingredient appears our dataset. Table 1 presents the ten most used ingredients:

Rank	Name	Number of appearance
1	salt	4527
2	pepper	3176
3	sugar	3081
4	onion	2819
5	egg	2282
6	butter	2229
7	flour	1999
8	water	1753
9	milk	1456
10	olive oil	1456

Table 1: Most used ingredients in our cleaned dataset

We obtain the first information about our weighted adjacency matrix: the maximum connectivity is 2448, it's for salt and pepper, which means that they appear 2448 times together in the 10000 recipes of the dataset; we get the mean connectivity: 0.0831, it represents the average number of recipes two random ingredients have in common. In the table 2, the 10 most widespread "couples" are displayed. Without surprise, they are the most common ingredients in our everyday life (salt, butter, sugar, egg...)

Rank	Ingredient 1	Ingredient 2	number of associations
1	pepper	salt	2448
2	onion	salt	1579
3	salt	sugar	1531
4	onion	pepper	1529
5	salt	flour	1383
5	salt	egg	1300
6	egg	sugar	1269
7	butter	salt	1240
8	flour	sugar	1216
9	egg	floor	1090

Table 2: Top ten connectivity between the ingredients

We decided then to investigate on the distribution of the number of relationships between the ingredients. Figure 5 shows the results. The number of connections is presented in logarithmic scale. We can see on the right of the figure the ten main "couples" presented just before, but such a high connectivity is uncommon while most of the couples of ingredients have a very few recipes in common: out of 3790881 possible couples, 3707043 have 0 recipe in common, 55910 have 1 recipe in common.

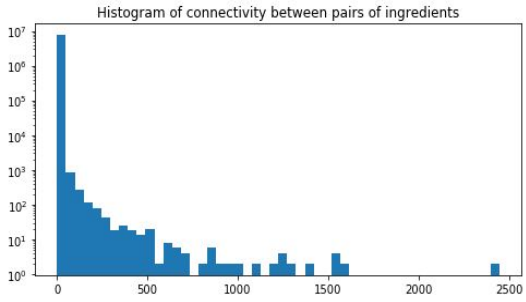

Figure 3: Distribution of the connectivity between pairs of ingredients

If we plot the log-log graph we obtain the following figure :

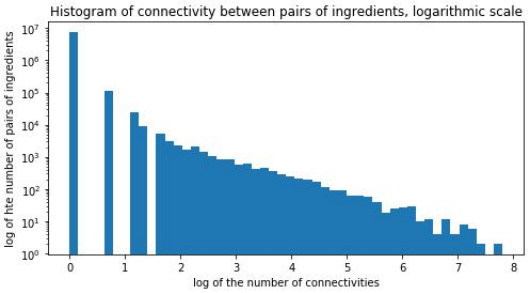

Figure 4: logarithm of the distribution of the connectivity between pairs of ingredients

We observe on the diagram that the distribution is similar to a power law, at least under a certain number of connections. For larger hubs, the connectivity tends to rise more.

The next step is to build an unweighted adjacency matrix, for this we take the matrix of connectivity and we apply a threshold, each connectivity (i.e. number of recipes in common) above the threshold will receive the value 1, 0 for the others. To fix the threshold, we use the distribution of connectivity between pairs, which is presented just before on figure 5. It led us to choose a threshold of 50 recipes in common to connect two ingredients. Playing with this threshold will be relevant for the last part of the project, to see the impact of the threshold on the created recipes. However, such thresholding leads to ingredients that are not connected to any other ingredients, because they do not appear 50 times with any ingredient. Thus, we need to prune our graph, i.e. to remove the nodes that are not connected to any other nodes. We investigated a little bit of the removed nodes, and it appears that this threshold also allows removing some outliers that are present a few times in the dataset.

Using this threshold and pruning the graph, we are going from a graph containing 2754 nodes and 715 edges, to one with only 131 nodes remaining, still with 715 edges.

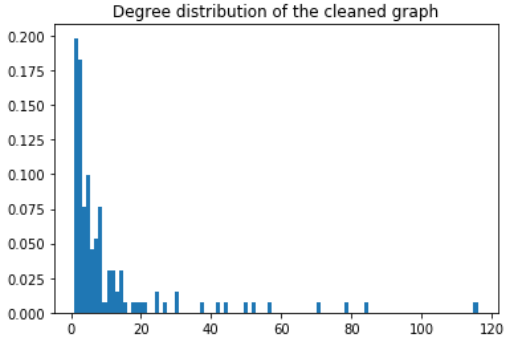

Figure 5: Degree distribution of the cleaned graph

Moments describe the distribution of degrees on the graph, the two first moments allow us to figure out the mean connectivity, and the spread of the distribution around zero. If we choose to remove the largest hubs, the moments are smaller then, as the hubs have the highest weights. We want to study the moments of the cleaned feature graph. We obtain the following values, first moment: 10.916, which means that on average, each node of the cleaned graph is connected to nearly 11 nodes, the second moment is 420.672. Using the two first moments, we can compute the standard deviation of degrees of the graph as follow :

$$std = \sqrt{M_2 - M_1^2} = 17.364$$

. To have an insight into the structure of the created graph, the figure 6 shows its visual representation.

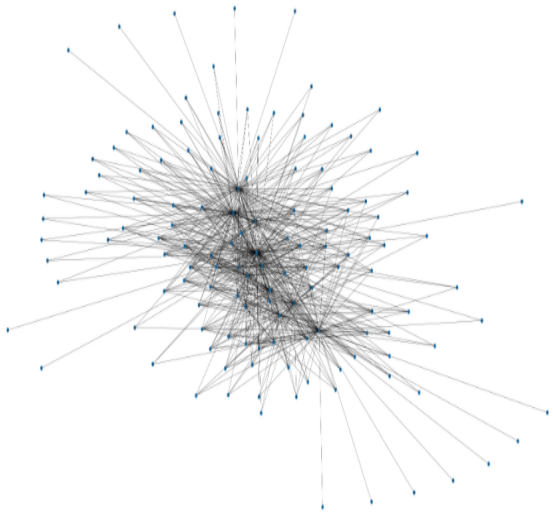

Figure 6: The graph after cleaning

4.2 Playing with the threshold

In the previous part, we used a threshold of 50 to create the adjacency matrix. That was a bit restrictive since a pair of ingredients should have at least 50 recipes in common to appear in the adjacency. This could make us lose classical associations, like for example fish and rice, indeed it's good sense to say that these elements fit well together but we are not certain if they appear together in 50 recipes. Then it may be relevant to lower the threshold. We will study the diameter too. It can be found thanks to the powers of the adjacency matrix. For a connected graph, we compute the sum of the powers of the adjacency matrix, from 0 to N, where N is the first power for which the sum matrix has no null element. N is then the diameter. But we decided to use networkx as the computation of diameter was already implemented in the library.

With a threshold of 50, we note that we have 715 edges among the 2754 nodes, and 2624 connected components, the matrix is sparse! We also remind that we cleaned this matrix from elements without any edge. There are then only 131 nodes remaining, they form a connected component, with a diameter of 3 (obtained thanks to networkx). We want to see the importance of the hubs into this matrix, to know this, we chose to remove the 12 biggest hubs (sugar, salt, eggs...) corresponding to the 12 ingredients that appear in more than 1000 recipes. This leads to a matrix of 119 nodes (131-12), 89 edges and 69 components. This allows us to see the importance of the hubs and how our food is based on these few ingredients.

To be less restrictive, we lower the threshold and take 5 instead of 50. There are now 8834 edges among the 2754 nodes, with 1937 connected components. We prune this matrix, by removing the 0-degree nodes, there are 818 nodes remaining (instead of 131!), they form a connected graph of diameter 4. We again remove the 12 largest hubs, and there are 4467 edges remaining, forming 806 connected components.

In conclusion we observe that lowering the threshold lowers the dependence on the classical elements (the 12 hubs). With a threshold of 50, we lose 87.5% of the edges (from 715 to 89) removing these hubs, with a threshold of 5 we only lose 49.5% of the edges (from 8834 to 4467). Thanks to this we are more able to represent classical associations not involving any element of the hubs (a vegetable with a meat, a fruit with chocolate...). However, lowering the threshold brings up the outliers and it augments the noise level.

5 Creation of recipes using a graph approach

5.1 Creation of random recipes

To create random recipes based on this graph, we have designed the following algorithm:

- A first ingredient is selected randomly, and put in a list of ingredients
- A list of neighbours ingredient is created based on the ingredients connected to this first ingredient.

- One of the neighbours of the list is chosen randomly and added to the ingredients if it is not already in the list.
- The neighbours of this last ingredient that are not already present are added in the neighbours
- The two last steps are repeated until we have the desired number of ingredients in the list

Examples of ingredients that could (maybe) be used together to create recipes are presented below:

- 'bamboo', 'baby bella mushroom', 'apple slice', 'avocado'
- 'chorizo', 'bone pork chop', 'cheddar cheese garnish', 'cornichon garnish', 'beef base', 'bosc pear', 'cheese herb', 'baby carrot'
- 'baby spinach', 'avocado safeway', 'almond extract', 'baby carrot', 'baby corn'

We can observe that the created recipes are possible recipes. We can imagine easily some of these meals implemented in real life. The presence in the dataset of recipes from different traditions can also be seen, with ingredients like bamboo that probably come from Asian recipes. Some generated recipes are also mixing ingredients from different cultures, with a result sometimes convincing, sometimes not.

5.2 Creation of recipes using a given or two given ingredients

This first algorithm allows creating random recipes. However, if we want to create a recipe using a given element, we just have to use his index as the first ingredient in the previous algorithm.
For example, here are 3 recipes containing avocado :

- 'banana leaf', 'baby spinach', 'baguette slice', 'avocado', 'almond paste'
- 'apple', 'arugula', 'avocado', 'baby carrot', 'banana slice'
- 'almond extract', 'avocado', 'almond garnish', 'baker chocolate', 'bamboo'

The things are a little bit more complicated if we want to put more than one given ingredient in our recipe. To find what recipe to do with two ingredients, we used the Dijkstra algorithm to find the shortest path between the ingredients, and we completed the recipe with neighbours of the ingredients in the recipe. To increase the number of paths and possible ingredients, we worked on the not cleaned graph with a low threshold, accepting to have sometime no path between ingredients because of outliers.

5.3 Evaluation of the realism of our recipes

We wanted to implement an algorithm to give a score to our created recipes, to have an objective indicator of the likelihood of our generated recipes. Our first idea was to measure the distance between our created recipe and the nearest existing recipe. A recipe close to an existing one means that we have created a realistic recipe. If the distance is higher, it can mean that such a meal is or very strange, or revolutionary. Our first approach was to measure the euclidean distance between recipes. However, because of the dataset cleaning, most of the ingredients were deleted from all the recipes, and thus it was not a good indicator. Our second approach was to measure the number of common ingredients. We had slightly better results, but the cleaning of the dataset leads to not interpretable results.

6 Conclusions

Using the Recipe1M dataset, we were able, after a cleaning phase, to make the data exploitable to create a graph. The graph has been explored and its properties were analyzed. The connections in the graph were used to generate random recipes or recipes from a selected subset of ingredients.

The main difficulty relied on the cleaning of the dataset. Our approach was to try to clean the dataset using a natural language processing library. It allowed us to have insights on what is going on in the dataset, using a subset of well cleaned-ingredients, but not to evaluate properly our recipes, because of the disappearing of most of the recipes information during the process. A dataset cleaning by hand would be much more fastidious, but could be realisable on a subset of recipes, and would allow having a much better insight into the quality of the recipes we have created. In addition, our cleaning introduces a bias in the exploration of recipes, as ingredients with simple names have much more chance to pass through and not to be considered as outliers.

References

[1] Rupp, rebecca (2 september 2015). "a brief history of cooking with fire". national geographic. https://www.nationalgeographic.com/culture/food/the-plate/2015/09/02/a-brief-history-of-cooking-with-fire/. Accessed: 2020-01-09.

[2] Website of the dataset. http://pic2recipe.csail.mit.edu/. Accessed: 2020-01-08.

[3] Wikipedia page of cooking. https://en.wikipedia.org/wiki/Cooking. Accessed: 2020-01-09.