# FooDIS - Food.com Data, Information and Search

ANTÓNIO BEZERRA, GONÇALO ALVES, and PEDRO SEIXAS

This work deals with the retrievement, refinement, analysis and querying of datasets regarding the website **Food.com**. The end goal is to create a search system capable of answering non-trivial queries that are not feasible through the website. To achieve this, the data sources are explored in order to extract information about all the recipes and reviews. This data is then refined through a process described within this paper and finally, it's analyzed through exploratory techniques. Next, through the use of Solr, we define a collection and its respective documents and describe their indexation. With this, the previously identified retrieval tasks are answered and evaluated according to their precision and recall.

Additional Key Words and Phrases: datasets, food, reviews, recipes, information processing, information retrieval, solr, indexing, statistical analysis

**ACM Reference Format:**
António Bezerra, Gonçalo Alves, and Pedro Seixas. 2021. FooDIS - Food.com Data, Information and Search. 1, 1 (December 2021), 9 pages. https://doi.org/10.1145/1122445.1122456

## 1 INTRODUCTION

Food is a necessity to us as humans. What started as simple as hunting and gathering developed side by side with humanity itself, being refined by the discovery of fire and preservation methods. Food is also a great way to identify cultures and to connect friends and family over a dinner table. As such, it made perfect sense for us to take a deep dive into this timeless topic, especially since the pandemic has increased the amount of cooking and baking people do in their homes (social media was booming with content related to this topic). Hence, the goal of this paper is to describe a search system that can help every level of chef obtain a recipe in an easier and faster way. This paper is split into three parts. The first, 2 , starts by describing the datasets and their refinement. Their quality is also assessed through exploratory techniques and some modifications are then performed, to obtain a final version of the dataset. This exploratory analysis allows us to have a better understanding of its values, distributions and patterns. Next, a conceptual model is presented to represent the final version of our dataset. This model depicts all the entities present in our domain, as well as the associations between them. Finally, some retrieval tasks are identified in order to represent some of the possibilities our final product is expected to have. The second part, 3, starts with a description of our collection and their documents, as well as their indexation. Then, some of our previously identified retrieval tasks are then answered and the results are evaluated, reflecting the quality of our results.

Authors' address: António Bezerra, up201806854@up.pt; Gonçalo Alves, up201806451@up.pt; Pedro Seixas, up201806227@up.pt.



## 2 DATA PREPARATION

Before finding a dataset, we started by defining what information we wanted to retrieve. Since food is such a broad topic, encompassing such things as diets, nutritional information, types of food (fruit, meat, dairy, ...), we felt that the best way to try and condense that much information would be with recipes. Finding sources on recipe information (i.e., ingredients, cooking time, nutritional information) is not difficult, as many sources already collect these properties, since this is often what people are looking for. As such, we came upon a Kaggle competition[3], where data, such as recipes and reviews, from the website Food.com[1] was collected. This was a good source since the dataset was easy to understand and the dataset was being actively maintained, as we can see from the usability score. After a brief analysis of the datasets, we concluded that both these datasets would result in a challenging but interesting project, not only by their size but by the information they contained. They have shown to be two very complete datasets, especially the recipes' dataset which had information not only about ingredients or instructions but also about calories and macro-nutrients.

Technical details:

- recipes
  - Size: 704.21 MB
  - Columns: 28 columns (16 textual, 8 numerical,2 dates)
  - Rows: 523 thousand
- reviews
  - Size: 496.1 MB
  - Columns: 8 columns (2 textual, 1 numerical ,2 dates)
  - Rows: 1.40 million

### 2.1 Data Pipeline

After this preliminary analysis, the first step was to extract users, categories, keywords, ingredients, and images into their separate tables. Besides being good practice from a database standpoint, this also allowed us to more easily explore the data and eliminate redundancy from our data. The next step was the cleaning of the data. To achieve this, we removed duplicate values, irrelevant columns and some outliers. Since a user can be a recipe and review author, when extracting the users from the original files, various duplicates were found and then promptly removed. The ingredients' quantities were removed since there were inconsistencies and because in the instructions the user can have a pretty good notion of how much of each ingredient. The outliers removed concerned themselves with some absurd caloric values (>50.000), and as such, it made no sense for them to belong in our dataset. After this, we proceeded to normalize some columns, such as dates or times, as the original file had them with special encodings. Finally, we thought it best to create a new column with the URL to the webpage of the corresponding recipe. The pipeline depicted below summarises the described process.

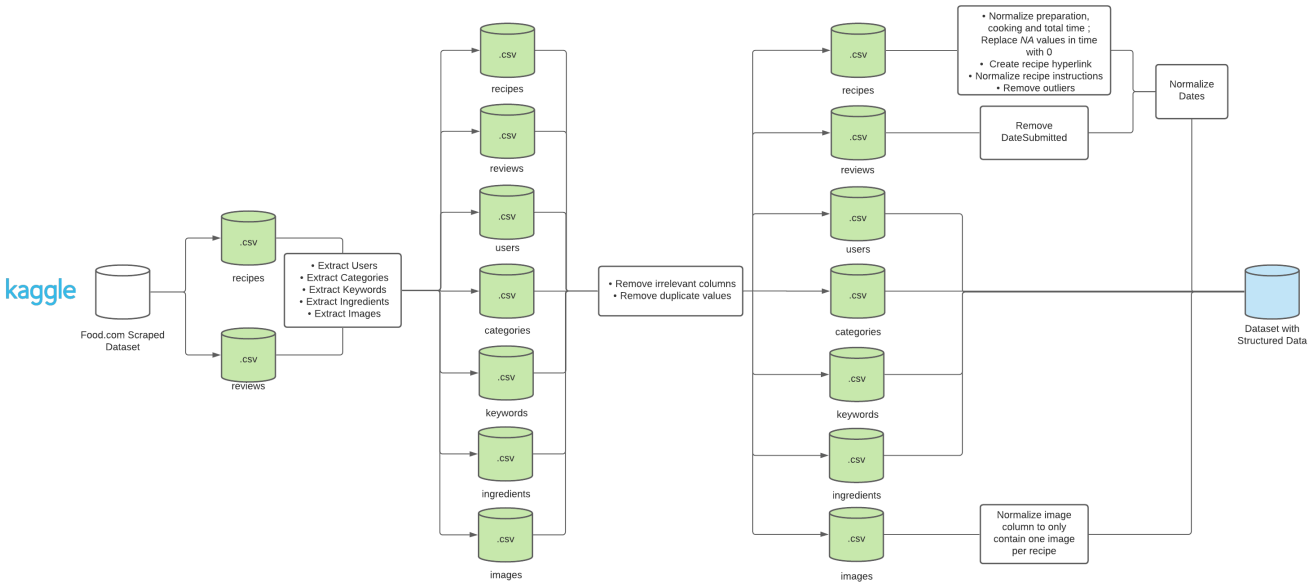


Fig. 1. Final Pipeline

## 2.2 Dataset Characterization

Our dataset can be mainly divided into two parts, the **recipes** and their **reviews**. To understand data patterns and assess if they are semantically expected, we made some exploratory analysis.

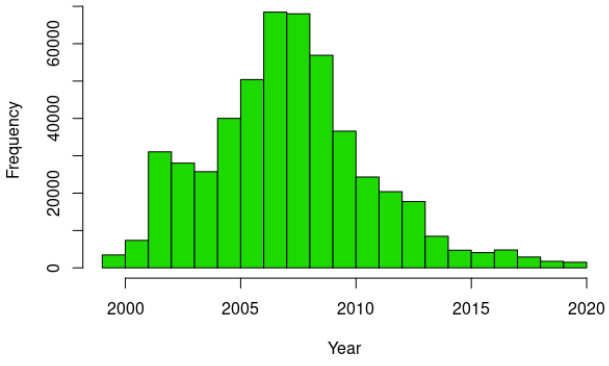


Fig. 2. Recipes Histogram

As for the **recipes**, our dataset has more than 500 thousand entries, published in more than 20 years. Looking at 2 we can see that the number of recipes uploaded to **Food.com** peaked 14 years ago, in 2007, with more than 60 thousand recipes being uploaded in a single year. This was due to the website's popularity at that time. After that, we can see a significant decrease in the number of recipes uploaded to the website, being currently on an all-time low, according to this dataset. This doesn't mean that the website has no current active users since there can be many recipes uploaded to the website that are not being scrapped by the creator of the dataset.

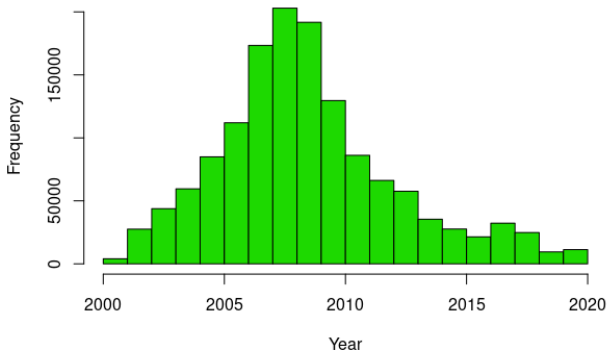


Fig. 3. Review Histogram

When talking about **reviews**, this dataset contains over 1.4 million reviews. The graph (3) looks very similar to the **recipes**' one, even though the frequency values are a lot higher which is expected since the reviews are scrapped from the recipe's page and the recipes can have tens or even thousands of reviews.

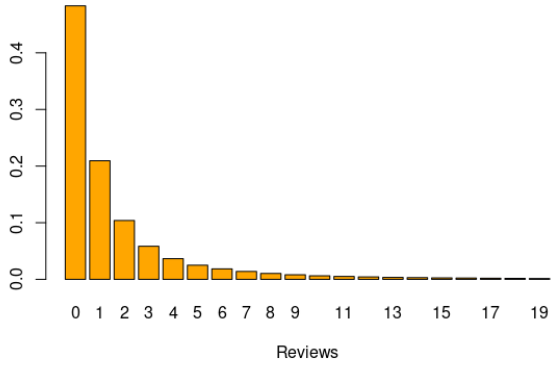


Fig. 4. Reviews per Recipe Histogram With 0

As we can see from Figure 4, more than 50% of the recipes don't have any reviews. To get this graph, we needed to filter the recipes, as some had more than a thousand reviews, being undeniable outliers. Without those recipes, the resulting graph shows a clear exponential decrease in the percentage of recipes, as their number of reviews increase. To get a better understanding of how was the exponential pattern, the recipes that did not have any reviews were removed, temporarily, to create a graph in Figure 5.

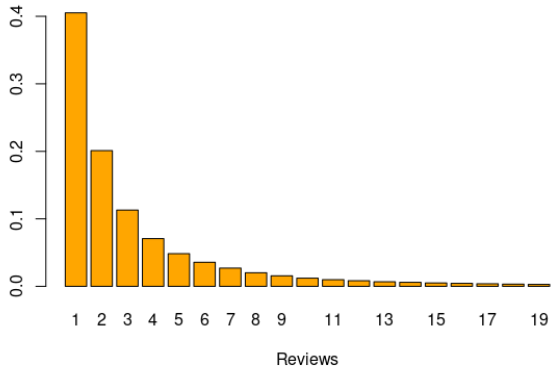


Fig. 5. Reviews per Recipe Histogram Without 0

Even without the recipes that did not have any review, we could see the exponential decrease. This was also verified if we filtered out the recipes with 1, 2, 3, 4 and so on, which resulted expectedly.

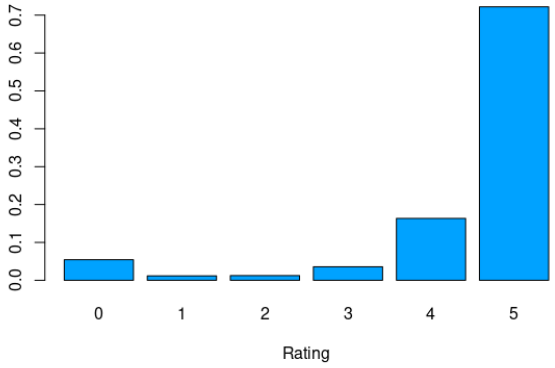


Fig. 6. Rating Histogram

After analysing the number of reviews, we thought it was interesting to find out what was the rating of these reviews to see if they are mostly positive or negative. With the graph in Figure 6, we can see that the number of positive reviews is significantly higher than the negative ones. One curious statistic is the fact that there are almost no reviews with 1, 2 or 3 stars, indicating that the users tend to use the extremes when reviewing, either positively or negatively.

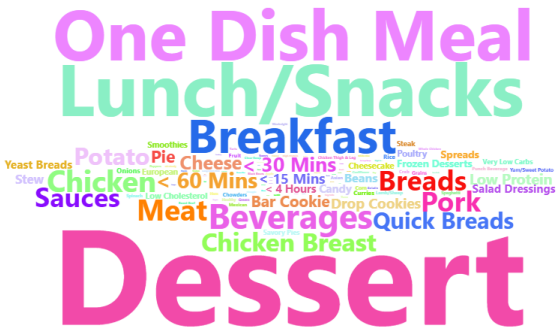


Fig. 7. Categories Word Cloud

After all the analysis in the reviews dataset, we found out that it would be interesting to know, since most of the reviews are positive, which categories of recipes were the most popular. For that, we made a word cloud (a great type of graph to measure the frequencies of certain words). In this word cloud, 7, we have some categories that are more frequent than the others, such as "Dessert" or "Lunch/Snacks". This statistic has just confirmed the thoughts at the start of this project, that there would be a lot of recipes about cakes and other traditional dishes. There are over 300 categories in this dataset, so this word cloud can not demonstrate the full list of categories.

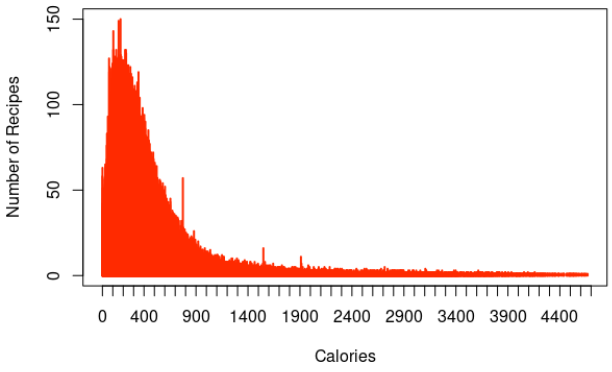


Fig. 8. Calories Per Recipe

With the categories in mind, another statistic that we could study would be the total calories of the recipes. As two of the main categories were "One Dish Meal" and "Lunch/Snacks" we already expected to have the highest frequency at around 300 calories, which is the normal calorie count for a complete meal and the graph in Figure 8 is demonstrating just that. The higher calorie recipes are mainly desserts or recipes with a high number of portions.

## 2.3 Data Domain Conceptual Model

The conceptual data model describing the collected data can be found below:

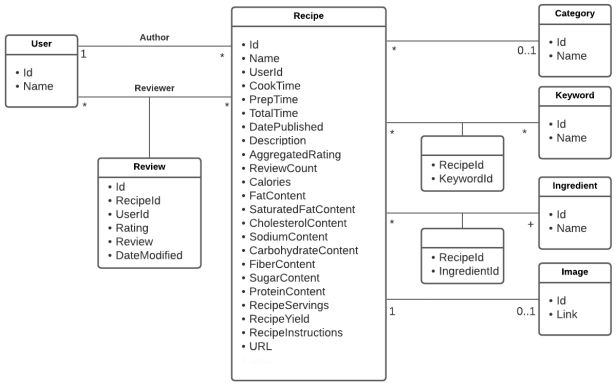


Fig. 9. Data Conceptual Model

The **Recipe** class is in the centre, as it is the main focus of our dataset. A recipe has an Author, can have a Category and an Image. It can also contain several Reviews, Keywords and Ingredients.

**Recipe** has the following attributes:

- **Id** : Unique recipe ID;
- **Name** : Recipe name;
- **UserId** : Recipe's author ID;
- **CookTime** : Time it takes to bake the recipe, in seconds;
- **PrepTime** : Time it takes to prepare the recipe, in seconds;
- **TotalTime** : Time it takes to prepare and bake the recipe, in seconds;
- **DatePublished** : Date that the recipe was published in the website;
- **Description** : Description of the recipe;
- **AggregatedRating** : Review ratings mean;
- **ReviewCount** : Number of reviews;
- **Calories** : Number of calories;
- **FatContent** : Fat in the recipe, in grams;
- **SaturatedFatContent** : Saturated Fat in the recipe, in grams;
- **CholesterolContent** : Cholesterol in the recipe, in miligrams;
- **SodiumContent** : Sodium in the recipe, in miligrams;
- **CarbohydrateContent** : Carbohydrate in the recipe, in grams;
- **FiberContent** : Fiber in the recipe, in grams;
- **SugarContent** : Sugar in the recipe, in grams;
- **ProteinContent** : Protein in the recipe, in grams;
- **RecipeServings** : Number of servings in the recipe;
- **RecipeYield** : Recipe yield;
- **RecipeInstructions** : Instructions to follow the recipe;
- **URL** : Food.com url;

Each **Review** is an association class between an author and a recipe and has the following attributes:

- **Id** : Unique review ID;
- **RecipeId** : Recipe's ID;
- **UserId** : User's author ID;
- **Rating** : Rating given in the review, between 0 and 5;
- **Review** : The actual review text;
- **DateModified** : Date that the review was made, or last modified;

## 2.4 Search Tasks

With the finalized dataset, different types of information can be queried. Some interesting retrieval tasks identified are:

- What desserts can I make?
- What can I make in a short time (less than 1h)?
- What can I make with chicken and lime?
- What are the highest-rated recipes?

the

# 3 INFORMATION RETRIEVAL

Information Retrieval is the process of retrieving a set of documents over a collection that best suit a given information need. The most common retrieval task is when a user specifies their information need through a query that is matched against all the documents and matching results are ordered by how relevant the system deemed them. The evaluation of the results is done in two ways: the first only performs matching on key fields, the second tunes the weights, by boosting more relevant search fields. The top 10 results for each query will be analyzed via their Recall, Precision at 10 and Average Precision values. Precision at n measures the precision level at a lower recall level ( n < collection size). Because Precision at n is a measure that is easily influenced by the number of relevant documents, Average Precision will be used to complement it. Average Precision is defined as an average of the precision values obtained for the set of top n documents existing each time a new relevant

document is retrieved. Since it does not depend on recall, it is considered a more stable measure. Recall, on the other hand, measures the fraction of retrieved relevant documents, i.e., what proportion of all the relevant documents has been retrieved. Calculating recall accurately requires comparing the set of retrieved documents with the entire collection, which is impossible in many cases, therefore it is possible to evaluate recall over a subset of the whole collection, using pooling.

### 3.1 Collection and Indexing Process

A recipe defines a document in our system. As there will be no other type of documents, a single collection is to be queried upon. To achieve this, we utilized the recommended tool, Solr[2]. Previously, we had described the way we transformed our two data sources into a more database-like structure. However, this structure is not appropriate to use with the selected tool. As such, we had to adapt our previously created dataset to create the single required collection. The changes made were:

- Join of recipes, users, ingredients and keywords - This eliminated the necessity of doing a join in query time, severely reducing the search time of our queries;
- Join with reviews - This way, we can easily access the recipe it is associated with, again eliminating the necessity of a join in query time. To achieve this we needed to concatenate the reviews' text with its author and rating so that we could have a string with all the necessary information;
- Added suffix string "T00:00:00Z" to dates - This is the format of a date in Solr;

As per the indexation of our document, not every field is expected to be searched by a user. Therefore, our first task was to identify which fields would be relevant to the user of our search system. These fields were the recipe's name, description, category, keywords, ingredients, author name, nutritional information, cooking time, time of posting and its reviews. For textual data, such as keywords or ingredients, a default Solr type, *text_en*, was used which already had the needed filters. These filters were:

- Stop word filter, discarding tokens that are on a given stop words list, such as "a" or "an";
- Lower case filter, preventing non-matches due to casing mismatch;
- English Possessive Filter, removing singular possessives (trailing 's) from words;
- Porter Stem Filter, removing the most common morphological and inflexional endings from words in English, like "jumping" to "jump";

At query time, an additional filter is added to the query, the synonym graph filter, which maps single or multi-token synonyms, producing a fully correct graph output. This is useful when searching terms that have synonyms capable of returning better results, such as "oven" and "bake", which ultimately point to the same thing.

We also used a custom type for the recipe's Name and Category fields. Since our goal might eventually be to build a user-friendly interface for our search system, having the ability to search for parts of words is useful for returning partial results. To achieve this we used a EdgeNGram filter, that creates tokens for parts of the word starting at the beginning. Table 1 specifies the added type.

The schema was defined using Solr's Schema (REST) API. The documents were indexed using Solr's Post tool by providing an input JSON array file. Table 2 specifies all fields used in the schema.

| Name | Tokenizer | Filters |
|---|---|---|
| title | StandardTokenizerFactory | ASCIIFoldingFilterFactory<br>EnglishMinimalStemFilterFactory<br>LowerCaseFilterFactory<br>EdgeNGramFilterFactory |

Table 1. Custom field types

| Field | Type | Indexed |
|---|---|---|
| Name | title | Yes |
| Date | date | Yes |
| Description | text_en | Yes |
| Images | uiString | No |
| Category | title | Yes |
| Keywords | text_en | Yes |
| Ingredients | text_en | Yes |
| AggregatedRating | float | Yes |
| ReviewCount | integer | Yes |
| Calories | float | Yes |
| Servings | integer | Yes |
| Yield | text_en | Yes |
| Instructions | text_en | Yes |
| URL | uiString | No |
| AuthorName | text_en | Yes |
| Review | text_en | Yes |

Table 2. Document schema fields

### 3.2 Use

This section will explore some features of Solr while answering some of the proposed queries. For each search task the information need, the corresponding query, the top 10 results, their relevance and performance are presented. As our original datasets have about half a million recipes and a million reviews, we will work with a smaller subset of 100 recipes and their respective reviews.

#### *3.2.1 Information Need 1 – Person that has chicken in the fridge and wants to search for recipes that use it.*

With this information need, we want to find all the recipes that contain chicken. A perfect system would only retrieve the recipes that contain some part of the chicken (breast, thighs, ...), however, some recipes may contain chicken broth or recommend accompanying with chicken, which might make the query harder. The chosen query is CHICKEN. For search boosting, documents where "chicken" appears in the title will have their score boosted relative to those where the search term appears only in the category or the instructions, and these will have their score boosted relative to

those documents where "chicken" appears only in the description, keywords and ingredients. The configuration and results for this query are presented below in Tables 3 and 4, respectively.

| | Regular | Boosted |
|---|---|---|
| Query | chicken | chicken |
| Parameters | qf: Name<br>Description<br>Category<br>Keywords<br>Ingredients<br>Instructions | qf: Name^10<br>Description<br>Category^2<br>Keywords<br>Ingredients<br>Instructions^2 |

Table 3. 'chicken' query parameter configuration

| | Regular | Boosted |
|---|---|---|
| **Rank** | **Recipe** | **Recipe** |
| 1 | Chicken Cacciatore | Chicken Cacciatore |
| 2 | Awesome Garlic Wings | Hot Chicken Sandwiches |
| 3 | Tom Kha Gai, Thai Coconut Chicken Soup! | Grilled Chicken Fettuccine |
| 4 | Grilled Chicken Fettuccine | Baked Rice and Chicken |
| 5 | Feta Topped Broiled Chicken | Feta Topped Broiled Chicken |
| 6 | Chicken Cutlet Parmesan With Tomato Sauce | Chicken Tomato Basil Soup |
| 7 | Awesome Chicken Noodle Soup | Elegant Stuffed Chicken Breast |
| 8 | Thai Satay Chicken Skewers | Awesome Chicken Noodle Soup |
| 9 | Hot Chicken Sandwiches | Thai Satay Chicken Skewers |
| 10 | Baked Rice and Chicken | Chicken Cutlet Parmesan With Tomato Sauce |
| Recall | 1 | 1 ($\Delta = 0\%$) |
| P@10 | 1 | 1 ($\Delta = 0\%$) |
| AP | 1 | 1 ($\Delta = 0\%$) |

Table 4. 'chicken' query top 10 results

In total, 30 recipes were retrieved. Both approaches achieved a Recall of 1, precision at 10 (P@10) of 1 and average precision (AP) of 1.

*3.2.2 Information Need 2 – Person that has just bought a new oven and wants to use it, but only has 1 hour to make something.*
For this information need, we want to find all the recipes that use an oven and have a total time (preparation + cooking) of 1 hour or less. The chosen query is OVEN. For search boosting, documents where "oven" appears in the keywords will have their score boosted relative to those where the search term appears only in the instructions, and these will have their score boosted relative to documents where "oven" appears only in the description. We also added a range filter, to limit results to the desired total time. Besides field weights, this query also takes advantage of an entry in our synonyms file that was used in the Boosted system. We define "bake" as a synonym for "oven", since it is a term frequently used in recipes that use ovens. The configuration and results for this query are presented below in Tables 5 and 6, respectively.

| | Regular | Boosted |
|---|---|---|
| Query | oven | oven |
| Parameters | qf: Description<br>Keywords<br>Instructions<br>fq: TotalTime:[* TO 3600] | qf: Description<br>Keywords^5<br>Instructions^2<br>fq: TotalTime:[* TO 3600] |

Table 5. 'oven' query parameter configuration

| | Regular | Boosted |
|---|---|---|
| **Rank** | **Recipe** | **Recipe** |
| 1 | Pizzcuit | Jim Dandies |
| 2 | Jim Dandies | Breakfast Egg Nests |
| 3 | Elegant Stuffed Chicken Breast | Quick and Easy Beer Bread |
| 4 | Breakfast Egg Nests | Buffalo Chips (Cookies) |
| 5 | Quick and Easy Beer Bread | Mexican Cornbread |
| 6 | GAL Muffins | My Favourite Sponge Topping for Fruit |
| 7 | Buffalo Chips (Cookies) | Elegant Stuffed Chicken Breast |
| 8 | Jim's Sweet Nuts | Witches Fingers (Cookies) |
| 9 | It's a Spicy-A Meatball | Jalapeno & Shrimp Poppers |
| 10 | Chewy Chocolate Chip Cookies (Ww) | Jim's Sweet Nuts |
| Recall | 0.73913 | 1 ($\Delta = 26.087\%$) |
| P@10 | 1 | 1 ($\Delta = 0\%$) |
| AP | 1 | 1 ($\Delta = 0\%$) |

Table 6. 'oven' query top 10 results

In total, 17 recipes were retrieved by the regular system and 23 were retrieved by the boosted one. This difference in search results is largely due to the synonyms file, which widened the search in the Boosted system. Both approaches achieved a P@10 and AP of 1, while having a Recall of 0.73913 in the regular system and 1 in the boosted system.

*3.2.3 Information Need 3 – Get recipes that only need small bowls, not medium or big ones.*
For this information need, we want to find all the recipes that can be made using a small bowl. The query chosen is SMALL BOWL. For search boosting documents where the terms "small" and "bowl" appear together, with a max token distance of 4, and in the instructions will have their score boosted relative to those where "small bowl" appears only in the keywords. The configuration and results for this query are presented below in Tables 7 and 8, respectively.

| | Regular | Boosted |
|---|---|---|
| Query | small bowl | small bowl |
| Parameters | qf: Name<br>Description<br>Instructions | qf: Name<br>Description<br>Instructions<br>pf: Instructions^5<br>ps: 4 |

Table 7. 'small bowl' query parameter configuration

| | Regular | Boosted |
|---|---|---|
| **Rank** | **Recipe** | **Recipe** |
| 1 | Pear Gingerbread | Dill Pickle Soup |
| 2 | Potato Soup & Smoked Salmon Relish With Crispy Brie Crouton | Delicious, Easy, Restaurant-Quality Eggplant |
| 3 | Dill Pickle Soup | Feta Topped Broiled Chicken |
| 4 | Deep-Dish Peach Pie | Breakfast Egg Nests |
| 5 | Strawberries and Cream Salad Dressing | GAL Muffins |
| 6 | Delicious, Easy, Restaurant-Quality Eggplant | Grilled Spice-Rubbed Salmon |
| 7 | Feta Topped Broiled Chicken | Thai Satay Chicken Skewers |
| 8 | GAL Muffins | Chewy Chocolate Chip Cookies (Ww) |
| 9 | Chewy Chocolate Chip Cookies (Ww) | Deep-Dish Peach Pie |
| 10 | Breakfast Egg Nests | Strawberries and Cream Salad Dressing |
| Recall | 1 | 1 ($\Delta = 0\%$) |
| P@10 | 0.6 | 0.7 ($\Delta = 10\%$) |
| AP | 0.631331 | 0.889719 ($\Delta = 25.8388\%$) |

Table 8. 'small bowl' query top 10 results

In total, 12 recipes were retrieved in each system. In the regular system, ranks 1, 4, 8 and 9 were not relevant, while in the boosted system, ranks 5, 8 and 9 were not relevant . The regular system achieved a P@10 of 0.6 and a AP of 0.63, while the boosted system achieved a P@10 of 0.7 and AP of 0.89. Both approaches achieved a Recall of 1.

#### *3.2.4 Information Need 4 – Recipes that have reviews saying that it was easy to make.*

For this information need, we want to find all the recipes that have a review stating that it can be easily made by a cook. The chosen query is EASY TO MAKE. For search boosting, documents where the terms "easy", "to" and "make" appear together, with a max token distance of 3, will have their score boosted relative to those where those terms simply appear in the reviews. The configuration and results for this query are presented below in Tables 9 and 10, respectively.

| | Regular | Boosted |
|---|---|---|
| Query | easy to make | easy to make |
| Parameters | qf: Reviews | qf: Reviews<br>pf: Reviews^5<br>ps: 3 |

Table 9. 'easy to make' query parameter configuration

| | Regular | Boosted |
|---|---|---|
| **Rank** | **Recipe** | **Recipe** |
| 1 | Dill Pickle Soup | Dill Pickle Soup |
| 2 | Pale Tigers Butter | Shipwreck Ground Beef Stew for the Crock Pot |
| 3 | Shipwreck Ground Beef Stew for the Crock Pot | Tater Tot Casserole |
| 4 | Tater Tot Casserole | Chicken Cacciatore |
| 5 | Chicken Cutlet Parmesan With Tomato Sauce | Pale Tigers Butter |
| 6 | Delicious, Easy, Restaurant-Quality Eggplant | Cottage Cheese-Banana Breakfast Delite |
| 7 | Chicken Cacciatore | Chicken Cutlet Parmesan With Tomato Sauce |
| 8 | Jalapeno & Shrimp Poppers | Delicious, Easy, Restaurant-Quality Eggplant |
| 9 | Alli's Renegade Cocktail Sauce | Jalapeno & Shrimp Poppers |
| 10 | Cottage Cheese-Banana Breakfast Delite | Alli's Renegade Cocktail Sauce |
| Recall | 1 | 1 ($\Delta = 0\%$) |
| P@10 | 0.9 | 0.9 ($\Delta = 0\%$) |
| AP | 0.8307 | 0.901912 ($\Delta = 7.1212\%$) |

Table 10. 'easy to make' query top 10 results

In total, 37 recipes were retrieved by each system. Both systems retrieved only 1 irrelevant recipe, which was in rank 2 in the regular system and in rank 5 in the boosted system. With only one recipe not being relevant, both systems achieved a P@10 of 0.9, but as the boosted system retrieved the irrelevant one with lower rank, its average precision was higher than the regular system, 0.9 compared to 0.83. Both approaches achieved a Recall of 1.

#### *3.2.5 Information Need 5 – Get Peter J's recipes published between 2008 and 2012.*

For this information need, we want to find all the recipes that were authored by Peter J between the years of 2008 and 2015. The pooled subset did not have relevant numbers of recipes by the same author. Since retrieving all recipes authored by someone is a feasible task, we opted to use the **full dataset**. The chosen query is PETER J. For search boosting documents where tokens "peter" and "j" appears in the recipe's author with a max token distance of 1 will have their score boosted. The configuration and results for this query are presented below in Tables 11 and 12, respectively.

| | Regular | Boosted |
|---|---|---|
| Query | peter j | peter j |
| Parameters | qf: AuthorName<br>fq: Date:[2008-01-01T00:00:00Z TO 2012-01-01T00:00:00Z] | qf: AuthorName<br>fq: Date:[2008-01-01T00:00:00Z TO 2012-01-01T00:00:00Z]<br>pf: AuthorName^20<br>ps: 1 |

Table 11. 'peter j' query parameter configuration

| | Regular | Boosted |
|---|---|---|
| **Rank** | **Recipe** | **Recipe** |
| 1 | Cheese and Onion Stuffed Sausages | Cheese and Onion Stuffed Sausages |
| 2 | Creamy Chicken Pizza Topping | Creamy Chicken Pizza Topping |
| 3 | Prosciutto Breakfast Bake | Prosciutto Breakfast Bake |
| 4 | Lamb's Fry and Bacon | Lamb's Fry and Bacon |
| 5 | Thai Dried Squid (Calamari) | Thai Dried Squid (Calamari) |
| 6 | Smoked Oysters and Bacon | Smoked Oysters and Bacon |
| 7 | Red Currant Sauce | Red Currant Sauce |
| 8 | Healthy Greek Style Chicken Tacos | Healthy Greek Style Chicken Tacos |
| 9 | Slow Roast Leg of Lamb | Slow Roast Leg of Lamb |
| 10 | Quick Steamed Scallops | Quick Steamed Scallops |
| Recall | 1 | 1 ($\Delta = 0\%$) |
| P@10 | 1 | 1 ($\Delta = 0\%$) |
| AP | 1 | 1 ($\Delta = 0\%$) |

Table 12. 'peter j' query top 10 results

In total, both systems returned 793 recipes. Both approaches achieved a Recall of 1, P@10 of 1 and an AP of 1 since they both retrieved the relevant documents at first.

### 3.3 Evaluation

Overall, the results were very satisfactory. Average precision was always above 88% in the boosted system and Recall was always 1, meaning our top results were relevant and all relevant documents were retrieved. The boosted system improved metrics when compared to the regular one, meaning our optimizations had the desired effect. Query 3 showed the most significant differences, improving AP by 25% and P@10 by 10%. In the cases where AP and P@10 were the same between the systems, we observed that the boosted system returned subjectively better results first. However, our analytical methods could not objectively measure that improvement.

Concerning the precision-recall curve, queries 1, 2 and 5 had similar curves, with the precision at 1.0, until recall reached 1, causing a precision drop, since the extra documents retrieved are not marked as relevant. This was observed in both the regular and boosted systems, as can be see in Figure 10. The other two curves had significant changes between the regular and the boosted systems that can be seen in Figure 11 and Figure 12, for queries 3 and 4, respectively.

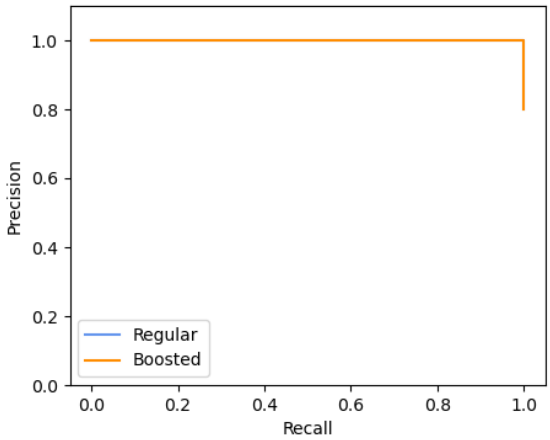


Fig. 10. Precision-Recall Curve for queries 1, 2 and 5

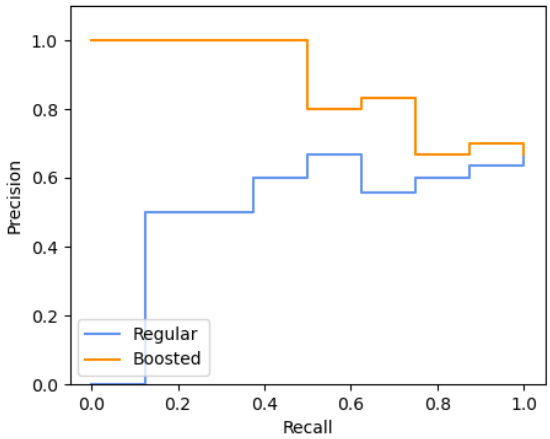


Fig. 11. Precision-Recall Curve for "small bowls"

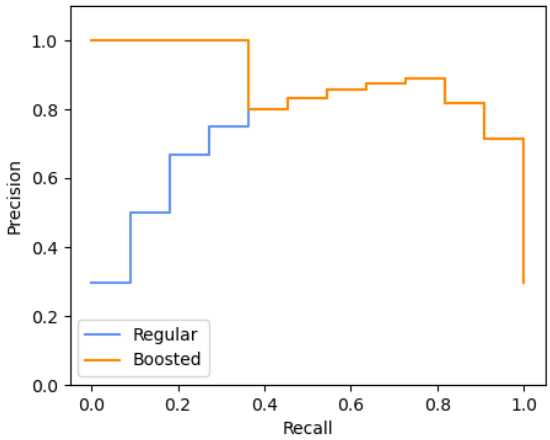


Fig. 12. Precision-Recall Curve for "easy to make"

In both these graphs we can see that the boosted system outperforms the regular one in terms of precision for low recall values. This means that the first documents returned by the boosted system are more relevant than the ones returned by the regular one. Therefore, our boosts are having the desired effect of pushing more relevant results to the top.

## 4 CONCLUSION

This paper addressed two distinct milestones in the construction of a system capable of answering complex queries about food recipes data. The retrieved data was cleaned, refined and conceptually described to represent the main classes and associations between them. An exploratory analysis showed patterns associated with the source of the data but also with the data itself. From this point, it was possible to identify information retrieval tasks that were later answered, using Solr and our schema. Results were positive for what would be expected by a user and were consistent with existing literature.

## 5 CITATIONS AND BIBLIOGRAPHIES

The following section serves to reference our citations and to list the materials used in this project.

Online citations: [4], [3], [2], [1].

## REFERENCES

[1] Discovery. 2021. *Food.com.* Retrieved December 7, 2021 from https://www.food.com/?ref=nav
[2] The Apache Software Foundation. 2021. *Solr.* Retrieved December 7, 2021 from https://solr.apache.org/
[3] irkaal. 2020. *Food.com - Recipes and Reviews.* Retrieved December 7, 2021 from https://www.kaggle.com/irkaal/foodcom-recipes-and-reviews
[4] Sérgio Nunes. 2021. *PRI 2021/2022.* Retrieved December 7, 2021 from https://web.fe.up.pt/~ssn/wiki/teach/pri/202122/index