Menu Analysis

## Problem Statement:

We are trying to conduct a an analysis between menus from different restaurants in Bangalore to try and make comparisons. We are trying to find the best method that can be used to find similar dishes across different menus.

## Setting up the data:

We are conducting this analysis to find similarities and differences between the dishes from two competing restaurants in terms of their description or ingredients between menus. To first set up the data, we manually collected from the publicly available online menus of the restaurants Bercos and Mainland china.

The menu for Bercos was found from their [website](https://bercos.net.in/menu). Since this menu was posted as an image, we used the the optical character recognition(OCR) implemented on the iphone camera to transcribing the text from the images of the menu into a .txt file.

The menu for Mainland was collected by downloading the pdf provided on their website

and then manually cleaning the characters in other languages and eventually converting it to a .txt file.

## Analysis:

To conduct the analysis we are mainly using the python dictionary data structures. By combing through all the items names we broke the entire menu

For example this is the first five items of both the dictionaries

| Fig.1. First five item of the Mainland menu dictionary |
| --- |
| Fig.2.First five items of the bercos menu’s dictionary |

By creating this dictionary it allows us to traverse through all the different items in the menu and access any items ingredients using the name of the item. To start formulating a plan to compare both menus we need to figure out a general idea of the entire data set. To do this, we break down the entire data set into dictionaries according to the categories listed in the original menu and count how many different dictionaries have been formed. From that we get the statistics in Fig.1.

|  | **Mainland** | **Bercos** |
| --- | --- | --- |
| **Number of items:** | 154 | 154 |
| **Number of categories:** | 24 | 23 |

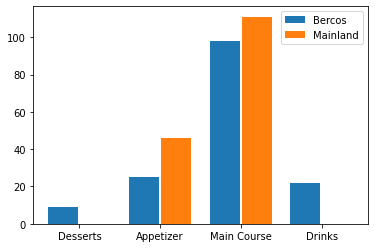
*Fig.1. High Level Statistics for both menus*

| *Fig.2. Item distribution from Mainland China’s original menu* | *Fig.3. Item distribution from Berco’s original menu* |
| --- | --- |

Fig.2 and Fig.3. are used to show the distribution of items in each of the categories listed. As we can see from these graphs there are too many categories to distinguish the number of items in each of them. Since each menu has over 20 categories we can reduce the categories by grouping them into bigger sections based on there similarities. This would make it more manageable to accurately understand the distribution.

| *Fig.4. Mainland main category breakdown* | *Fig.5. Berco main category breakdown* |
| --- | --- |
|

Manually looking at the different categories, we could see 4 very apparent groups. For Mainland, it was Appetizers, Jain items, Main course items that were not Jain and items that did not fit into any of those three: other. Alternatively, for Berco’s menu, there were Chinese and Thai items, desserts and drinks and items that did not belong to the other three: other. This breakdown was more clear and made it easier to view the distribution a little better.

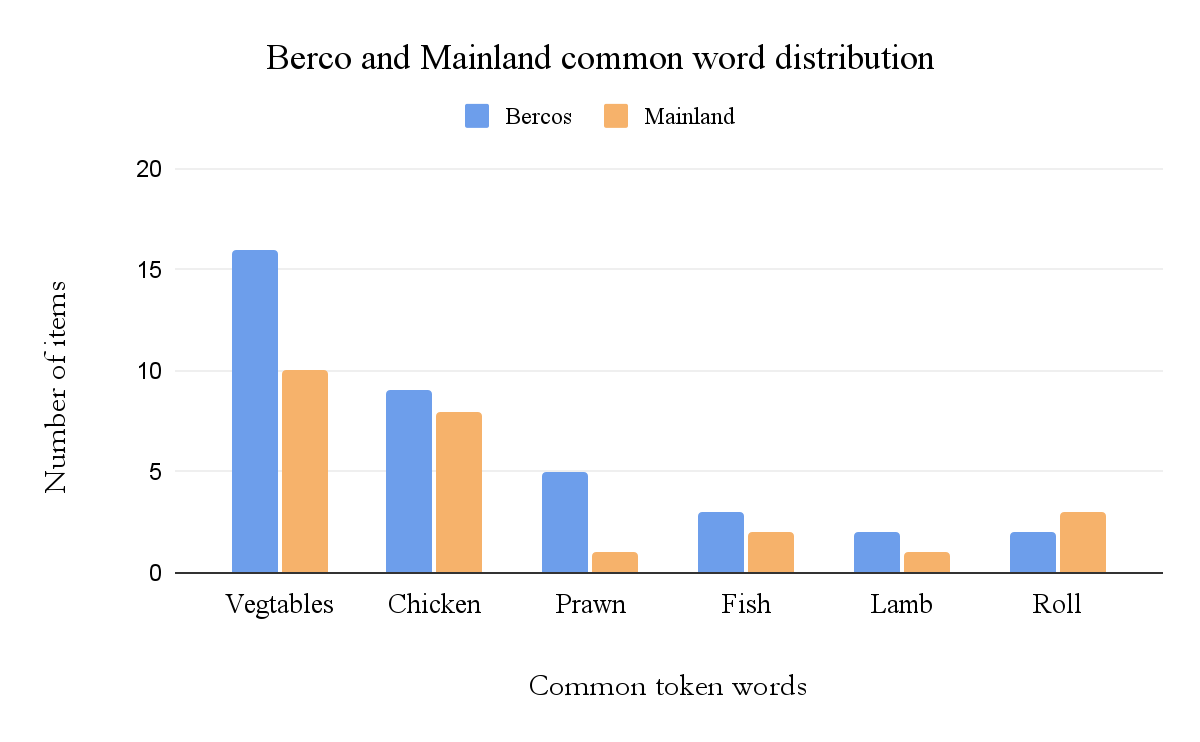


*Fig.6. Comparing number of items in each category for each menu*

One of the problems that we faced was that the categories from each of the restaurants were not the same, and therefore could not be compared with each other. We further separated the categories in Fig.4. And Fig.5. into more common groups like Desserts, Appetizers, Main Course items and Drinks. Since, both menus have the same categories we can compare them side by side. Another problem faced in this stage was that one of the menu(Mainland) was missing a couple of the categories.

As seen in Fig.6. Mainland’s menu did not have any items in the Desserts and Drinks category. One of the main reasons might be because the restaurant might have a separate dessert or drinks menu that we do not have access to.

For our purposes we decided to use the appetizer items and main course items and compare them further. To understand if there are any similar items or how to approach the comparison we used an excel sheet to manually compare them first.



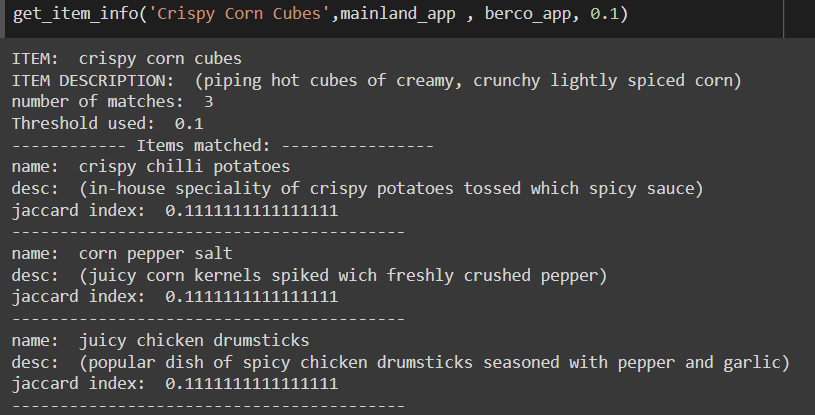
*Fig.7. Manually finding the common token words in each item on Google Sheets*

By manually comparing them we spotted some common tokens in all the items. These words were mainly roll, fish, prawn, chicken, lamb and vegetable. Since each token was present in both menu we could successfully use the Jaccardian Distance formula to an extent.

### Jaccard Index:

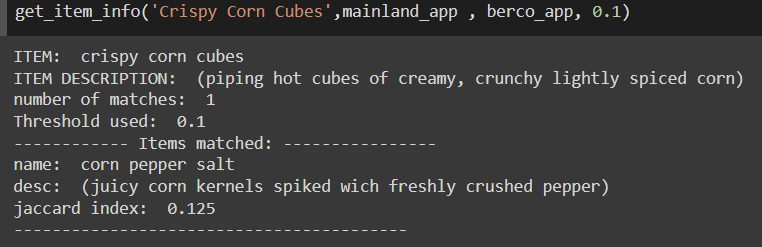
Jaccard Index or Jaccard similarity Coefficient is a method of comparison that can be used to compare word tokens. This method treats the data objects like sets and is defined as the size of the intersection of two sets divided by the size of the union.

To calculate the Jaccard index, we first started by breaking down the description into tokens for each item in the dictionary. While processing these tokens, we got rid of punctions so that the matching could be focused on just the words. When we first processed there were some issues that we came across.



*Fig.8. First iteration of the Jaccardian index implementation*

As we can see in Fig.8. the item we were matching was Crispy Corn Cubes and the function found 3 matches with a threshold of 0.1. This is a very low threshold, but for the purpose of this example we can use it to see a difference. When we closely look atthe matches found we can see that the item’s descriptions are not very similar. One is a potato dish, a corn dish and a chicken dish. This is caused because the function is matching words like “and”, “with”, “tossed”, etc. This makes it a problem because these words are present in the description for most of the items. To overcome this problem, we implemented stop words. When the tokens were being processed, we added a list of stop words like “and”, “with”, “in”, “for”, etc.

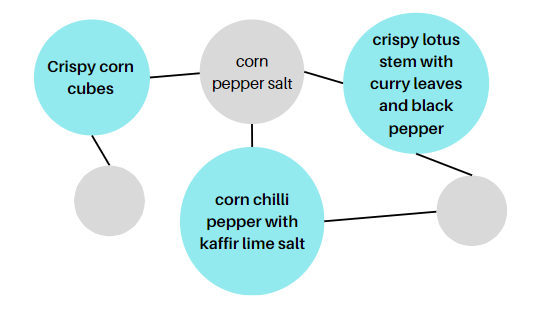


*Fig.9. Second iteration of the Jaccardian index with stop words implemented*

Now we can see in Fig.9. that the search is more refined after adding the stop words. The function only found 1 match with a threshold of 0.1 and the description matches a lot better to the item given.

## Next Step:

We are going to build a network connect the items together through branches. We want to build an interactive web page where the client(Chefs) can enter certain key words of different items and the restaurants names to figure out the most popular food on the market. Senior chefs can use this help restaurants to build their our menu or help them market their brand better.



*Fig.10. Prototype of the model*