Image Classification of Food Data with Nutritional Insights

Sameer Pasha(301406535), Archana Subramanian(301402868), Ravi Kiran Dubey(301406428), Vignesh Vaidyanathan(301398663)

Problem Statement

Food is an integral part of our life. There are two main groups of people - those who consume food without paying attention to the detrimental effects it may have on our bodies and those of who are quite health conscious and meticulous about their diet.

In this project, we aim to develop an application that can classify food items and would list down its nutritional components and recommend popular restaurants to try the food, famous recipes and providing the visualized output for the same which helps the users in making well-informed decisions about their food consumption.

Data handling and Pipeline

Nutrition Data:

The food image whose nutritional data we need is first scanned and identified. At this stage, a label for that food is generated. This nutrition data for this specific food label is identified and an insight is provided. The nutrition data is obtained from the internet and has the following nutritional information: Sugar, Sodium, Potassium, Carbohydrate, Fat and other such nutritional data.

Restaurant Data:

The food label obtained from the previous step goes through the restaurant data. This data is extracted from and is a subset of yelp dataset. The data contains food label, restaurant name and rating of restaurants that serve appropriate food. This data is extracted from business.json file of the yelp dataset. The restaurant suggestion for the food label is visualized by marking the geographical position of the restaurant.

RecipeData:

We provide top rated recipes for the food scanned. This data is scraped from the web **ON DEMAND**. For a specific food label, the best recipe, its name and image data is scraped and stored. This is then represented as an image card which redirects to the recipe link.

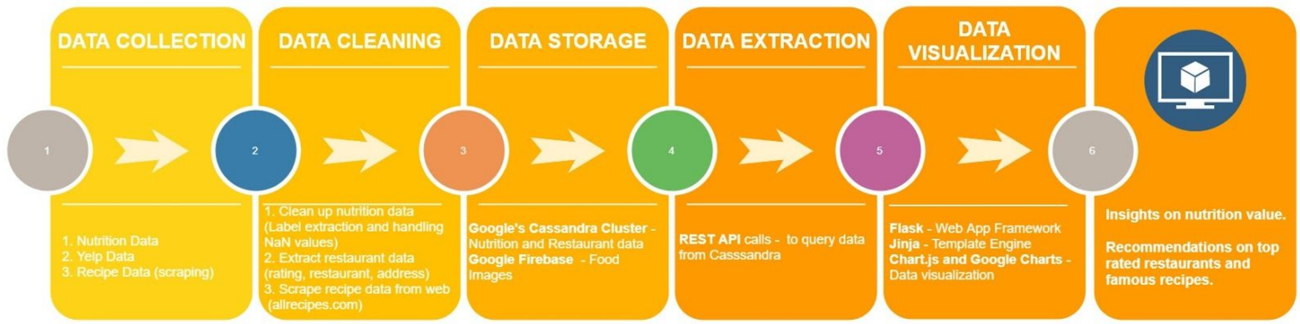


Fig.1. Pipeline Diagram

Methodology

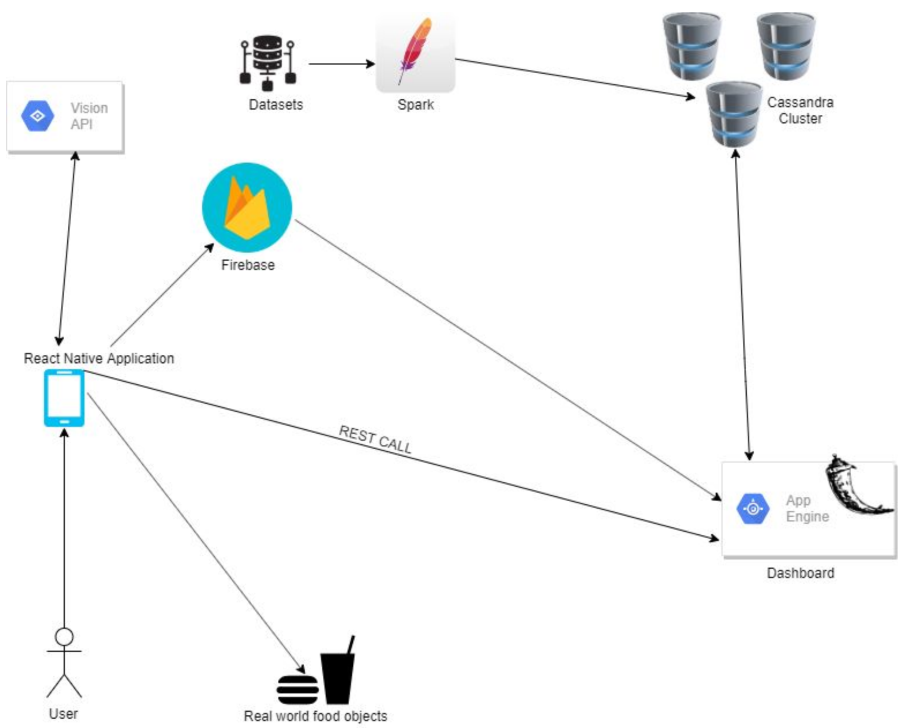


Fig.2. Architecture Diagram

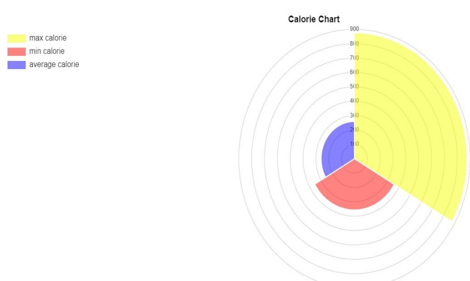
We have created a mobile app using **React Native** which supports both Android and iOS. The user can interact with the system using this app. The app lets the user take an image or use any image from their gallery.

The captured image is then uploaded to Google Firebase and sent to **Google Vision API** for smart label detection. **Firebase** is near realtime and it is ideal to use this for storing our image data as it can fetch images and send images at near real time speed.

Once we have identified our food item, we then send 3 **GET requests.** The system is hosted on **Google App Engine** which runs on a custom Flexible environment, which helps the application run on a compute engine VM with auto scaling enabled. This helps us handle extra traffic by automatically provisioning more VMs if required.

On Google App Engine, we use **Flask** to host the application. Once the data is received from the app, we then send this data to our **Google Cassandra Cluster**, which currently has 3 nodes and will automatically extend to 5 VMs if extra load is received.

We fetch relevant food related nutritional information and restaurants recommendation and also **scrape** the top rated dishes and pass this information to our visualizations page through our **Template Engine**, which is **Jinja** in our case. Finally, we get all the relevant insights and recommendations which we can monitor through our dashboard.



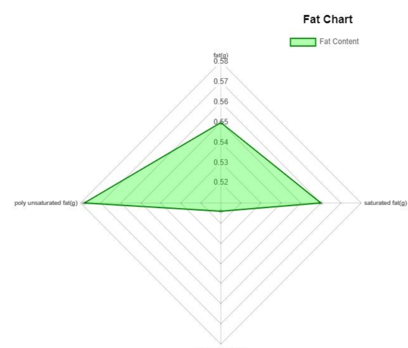


Fig.3: Calorie Chart

Fig.4: Fat Chart

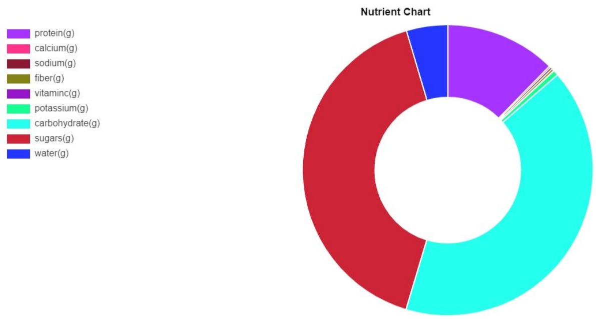


Fig.5: Nutrient Chart

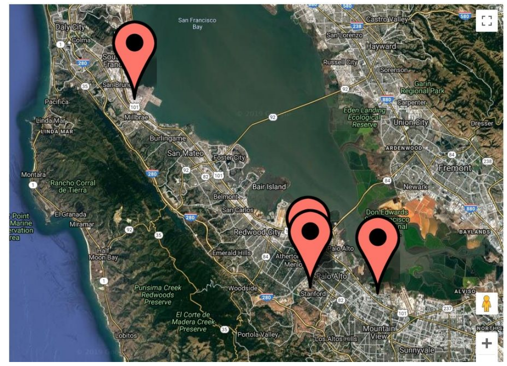


Fig.6: Popular Restaurants

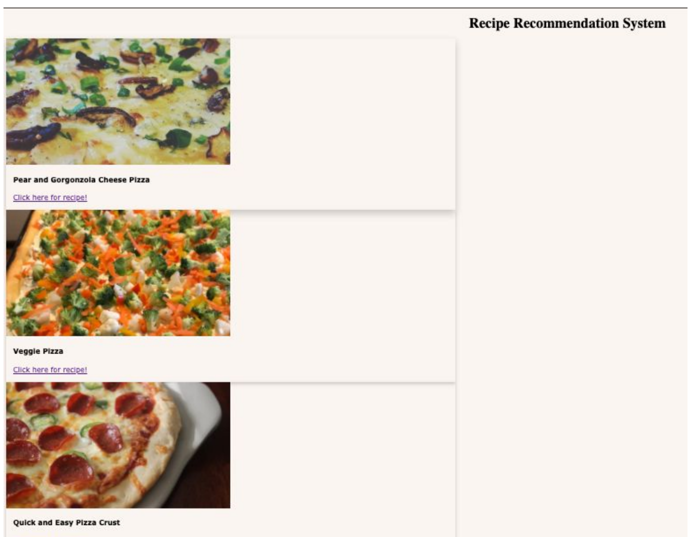


Fig.7: Recipe Recommendation

Problems

There were a lot of challenges that we faced while doing this project. Some of them are listed below:

- The first challenge was connecting to google's cassandra cluster through client drivers. This was resolved by adding new firewall rule for allowing TCP connections on port 9042.
- The other challenge was making cassandra work with Google App engine and webapp2. This is because cassandra cluster source has some c extensions which is not supported by the Google App engine. This was resolved by switching to a flexible environment where the application runs within docker containers on compute engine VMs and also lets us install any custom third party libraries.

Results

Given an image of any food item, we are able to classify the food item. Once the item is classified, we make 3 **GET** requests, where we pass the food label name. After receiving the food label, the following tasks are done:

1. Query Cassandra with the food label to get nutrients information.
2. Query Cassandra with the food label to get top rated restaurants information.
3. Scrape recipe data for the identified food label on-demand.

This application enables people to make an informed decision in their food consumption. It also enables them to learn how to prepare the food by providing them the food recipe. In addition to that, the application also acts like a recommendation system, providing the location of the highest rated restaurant for that corresponding food item.

4. We were able to manipulate the nutritional data to extract compositions of the food being scanned and provide a meaningful insight on it.
5. By picking out only meaningful data from the YELP dataset, we were able to provide recommendations on restaurants with the maximum rating and their corresponding location on maps using GCharts
6. On performing **ON-DEMAND scraping**, we are surfing excerpts of recipes which are most popular. Since we are not scraping the entire data, we are not storing a lot of information on the system, thus enhancing speed and reducing latency.

Future Scope

This project can be more helpful to the users if they have their own profile in the application. In this way the user will not need to search for his/her favourite dish or recipe every time. The user can just save all the food items and recipes to the account and use it whenever he/she wants.

This project can be further improved by showing insights on the origin and popularity of food items on a world map. In this way the user can choose his/her cuisine properly from a wide variety of choices.

Project Summary

We can thus conclude that with the data we infer from the application, the user is more aware of what he/she consumes and also recommends recipes and restaurants. SeeFood can shortly be said as an application that conveys to the user, valuable insights about the food he/she consumes and also provides most popular places to eat that specific food and prominent recipes for the same. It can be described as a system that is a combination of an analysis and recommendation systems.

Getting the data	On-Demand scraping, Yelp dataset, downloading nutrients dataset	2
ETL	NaN values, elimination of inappropriate attributes	2
Problem	Connecting to google's cassandra cluster, making cassandra work with Google App engine	3
Algorithmic work	Aggregating food values to get meaningful insights	1
Bigness/Parallelization	Google Cassandra cluster with auto scaling enabled, Google flexible environment with auto provisioning.	4
UI	React, Flask with Jinja, Google Chart, Chart.js, HTML	2
Visualization	Nutrient composition, Fat and Calorie amount, Restaurants recommendation, Recipes suggestion	2
Technologies	Apache Spark,Google Firebase, Google Cassandra Cluster, Flask, Jinja, React Native, Google App Engine	4

References

1. https://www.allrecipes.com/
2. https://www.chartjs.org/
3. https://www.yelp.com/dataset
4. https://docs.expo.io/versions/latest/
5. https://github.com/JscramblerBlog/google-vision-rn-demo
6. https://bl.ocks.org/syntagmatic/raw/3150059/

GIT:

https://csil-git1.cs.surrey.sfu.ca/spasha/food-visualizer

Tag: FINAL