Business environment

Digital Health Inc. have an existing calorie expenditure system in the market. Creation of calorie intake system provides the company with a full-scale calorie expenditure/intake solution. There are many similar applications in the market but most of them only provides the solution to either customer’s expenditure or intake of calories but not both. This creates a big opportunity for the company in the market. The calorie intake application requires an accuracy of 94.11 percent. The accuracy is much better than the accuracy of existing applications such as- calorie counter, Fitbit, Argus etc. The application also requires an auto-calibration system to detect the volume of the food, which is more accurate and easier to use. The user does not have to put in his finger or a coin to help the application measure the volume of the food. The accuracy and ease of use will work as a source of strength for the company. The market is filled with many applications that does the same thing. Our application has the potential to be most accurate in the market. Together with calorie expenditure application, we have the potential to deliver a full-scale calorie intake/expenditure application, that is not common in the market. We as a developer, has the responsibility to be 95 percent accurate in calorie detection, as many people, athletes and health organizations will depend on this application.

Missions, goals, and objective

Mission statement:

To create a healthier everyday life for many people.

Goal:

Build an android and iOS cloud-based calorie detection application with 94.11% accuracy and uses auto-calibration technique to measure volume of the bowl, for personal and commercial purposes within next 4 months.

Objectives:

The application will detect calorie with 94.11% accuracy which is much better than any existing product in the market. The existing products use a coin or a thumb to measure volume of the bowl whereas our product will use auto-calibration techniques. The application such as “Calorie Mama AI”, “Fitbit” requires more than 6Gb of space. Our application is mostly cloud-based, resulting in a fast and space-efficient application. The application will be used by health companies and individual users, aiming at a wide range of customers.

Business Model

How will we achieve 94.11% accuracy:

The system then uses image processing and computational intelligence for

food item recognition. The item recognition allows the system to be trained with single images of items. At the training stage, we use region proposal algorithms to generate candidate regions and extract the convolutional neural network (CNN) features of all regions. we perform region mining to select positive regions for each food category to generate set of candidate regions.

For each region, a classification score is computed based on its extracted CNN features and predicted

food names of the selected regions. Our experiments, conducted with the FooDD dataset, show an average recall rate of 90.98%, precision rate of 93.05%, and accuracy of 94.11% compared to 50.8% to 88% accuracy of other existing food recognition systems.

Cloud-based fast and space-efficient application:

Since fast response is one of the important parameters for the user who wants to eat the meal, certain heavy computational parts of the application are offloaded to the cloud. Hence, the processes of food recognition and calorie estimation are performed in cloud server. The application will be smaller size compared to similar existing applications in the market.

Auto-calibration:

We use deep learning to accurately train and classify the food object to its corresponding label. With the help of mobile sensors, the system in real-time, can gauge the distance that the user is standing from the food object. The system will then record this value and send it to the cloud along with the food photo captured. The image will then be processed in the cloud with the help of virtualization, and the results (including the calorie value and the food object label) will be sent back to the user on the mobile device. We have also proposed a new method whereby the application will assist the user in real-time for determining the ideal distance from which the user must capture the photo.

Information Environment:

Project Portfolio:

The project is required to finish within 4 months. Our company will spend most of their resources on this project. The project is top priority.

Long Term System Plan:

The application’s operating system is android or iOS, any new component’s UI must be from these libraries. The application uses single trained images of objects to train the model, any further consideration of increasing accuracy should be trained with single images of objects. The application needs to be lightweight and fast, so the heavy computation must be in the cloud. The Amazon cloud will provide it as Software as a Service.

Database Configuration:

Analytics & Reporting

Analytical Data Store

Batch Processing

Data Storage

Android & iOS

Real-Time Ingestion

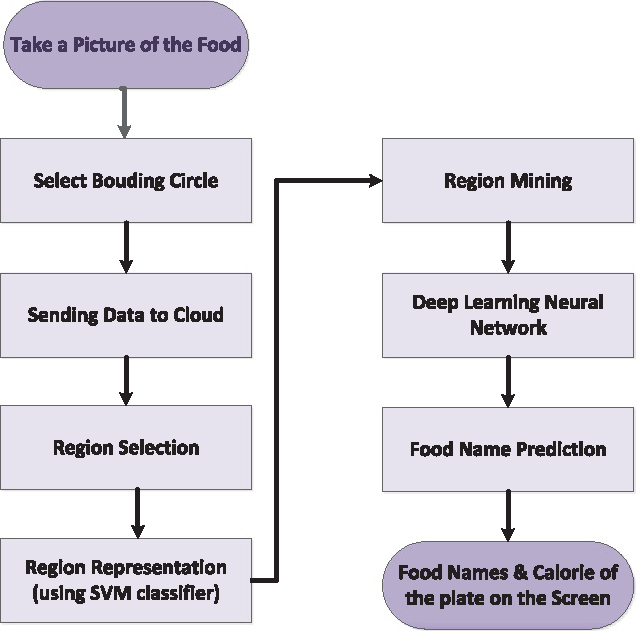
Stream Processing

Fig: Big data architecture for ML

The database will save the pictures taken from user’s phone and save it after training the system with that data. The user will have only access to their own photos and Software will only access the part of the database to match the user’s photo and calculate calorie from it. The data should be saved by each object and their sets. Each object’s set should contain all the object’s images from different customers. The access time should be low, as the system will know where to look, when asked for that object. For security reasons, the application of that user will only have access to that object’s images in the database.

System Processes:

The business activities the system needs to support are the means to get 94.11% percent accuracy while the whole computation takes place in the cloud. The result of the computation must be accurate and fast at the same time.

****   

|  |  |  |  |
| --- | --- | --- | --- |
| Food Name | Measure | Weight(grams) | Energy |
| Apple, with skin | 1 | 140 | 80 |
| Orange | 1 | 135 | 116 |

Fig: Different phases of food detection.

System operational rules and policies

Business rules:

1. Data security: The application must interact with cloud in a way that, application does not have access to that saved data. The application will save the data, conduct calculation on that data, show results securely without application intervention to that data. Data must be saved securely in the cloud.
2. Up time: Business must be up at least 90% of the time.
3. Ease of Usability: A new user should be able take photo and count calorie within 5 minutes of starting it. The User Interface must be intuitive, easy to use.
4. Subscription: The health companies must have paid subscription to use this application for many people.
5. Speed: The time between user taking a photo and the application showing the results should not be more than 2 seconds.
6. Maintainability: Each function must be described and documented so that, future maintenance and update are easier to build and deploy.
7. Meeting the demand: The application must be deployed in a serverless way, so that no user has to wait for accessibility of the application.
8. Mean time between failure: Mean time between failure should be more than 10 days.
9. Accuracy: The application should detect the foods and their quantity with 94.11%.
10. Reviews: The user reviews must be taken into consideration to improve the system.
11. Finish time: The application must be ready within 4 months.

Business policies:

1. Customer quality policy: The customer’s reviews must be value. Customer will be treated with respect all the time.
2. Compensation policies: If a paid subscriber or a health company encounters an error regarding calorie calculation, he/she must be compensated.
3. Data policy: Data must be securely saved and used only for calculation and machine learning purposes. These data will not be used for any other commercial purposes.
4. Social media: If you have a concern about the calorie count network, employees, or application users, we encourage you to reach out to us so we can address your concern! We ask that you refrain from posting harassing, disparaging, fraudulent, or offensive materials about other calorie counter users on social media.
5. Calorie counter API should not be used in connection with spyware, adware or malicious programs or code.
6. Calorie counter should be used for any purpose that violates any law or regulation or any right of any person.
7. Developers should not use calorie counter in any manner that adversely impacts the stability of “Amazon cloud”.
8. Developers should never sell, sublicense, or lease calorie counter API.

Addressing the policies and rules:

The data of the application will be saved in the cloud. Amazon cloud has security measurements to ensure security of the data. The cloud provides enough resources to calculate the result and showing it to the user within 2 seconds. The users will have restrictions on other user’s data. The codes will be documented as we write it to ensure easiness in maintainability. The application will be deployed in the cloud in serverless way so that it meets demands. The developers will not be able to put any malicious codes in the application as all the codes will be reviewed by the chief programmer before it gets included.

Operational Constraints:

Functional requirement:

The Register component should save all the information from the user and save it in the database. The user should be able to login using the previously given credentials. This component should allow user the ability to change their own credentials upon verification. The cloud component of the application will do the heavy calculation and provide the user with results. The results must be greater than or equal to 94.11% accurate within 2 seconds of time. The auto-calibration part of the application must calculate the quantity of the item accurately without requiring user’s coin or thumb. The convolutional neural network must detect the food with accuracy of 95%. The Results should be calculated from nutritional table. The results must be calculated in the cloud and presented to the user with out using resources from user’s device. There must be away to take the user review after each scan and add to the data, for the machine to learn. The application must save data securely and restrict users to access other user’s data. The nutritional insights must give insights based on each user’s food habits and health condition. The new recipes component of the system should suggest new ideas based on that user’s personal likings. There should a list of new recipes for users.

Conditions:

The application must follow cloud computing rules to use cloud computation. The computational parts must be completed in the cloud so that user device’s resources do not get used. The food detection technique must accurately detect items I difficult conditions such as bad lighting, uncommon shape of the bowl, state of the food etc. The quantity detection technique must use auto-calibration. The health organization can only use this application with paid subscription. Data should be saved securely, so that no user can see or use other user’s information. The cloud provided security measures must be taken to ensure safety. Databases will save the information after grouping together similar information to ensure speed of the application. The reviews must be incorporated to the wrong/unsatisfied results to improve our system. The UI must be pleasant and easy to use. The user should intuitively place first scan within 5 minutes of starting the application.

System Operational modes and states:



Data Storage

Cloud

Android & iOS

Wrong result and review

Region mining

Auto-calibration

CNN





Nutritional Table

|  |  |  |  |
| --- | --- | --- | --- |
| Food Name | Measure | Weight(grams) | Energy |
| Apple, with skin | 1 | 140 | 80 |
| Orange | 1 | 135 | 116 |

The blue lines in the figure shows how data is passed between each mode. The cloud component uses all the modes under it and gets the result to user and saves it in the database. The orange line shows if the user review is bad then that review is added and then the photo gets saved in the data storage. Now if another user posts a similar photo then the application will serve the result without any computation.

System Operational Quality:

Performance:

The time between user taking a photo and the cloud providing the results should not be more than 2 seconds. To achieve such efficiency, the application database must be grouped with some sort of key and stored in the database. The cloud must have enough resource allocated so that the computationally intensive tasks of convolutional neural network and auto-calibration happen very fast. The user login and registration need to complete in 1 second. In a second, the application must save the information securely in the cloud. The software must no hang or stop working in the middle user using it.

Compatibility:

The calorie counter is a mobile application. It will wok in Android and iOS operating system. As the user needs to take photo of the food few times a day, it is not convenient for the user to use laptop or desktop. The software will use DynamoDB of the Amazon cloud. The application will properly work on any version older than “Android Eclair” (2.0) and “iOS 4.0”.

Reliability and Security:

The proposed method’s accuracy is 94.11% which is more than any other calorie detection application in the market. The software gives the user, an opportunity to review on the result provided. The application will add the user review for the machine to learn. Even if the application gives a wrong result under adverse condition, it has the potential to make it right next time. The data used by the software will be securely saved in the cloud. Users will be restricted to access other user’s data. The user can rely on the results and the data security of “Calorie counter”.

Maintainability:

The codes of the application will be documented for the easiness of updates. The application architecture is modularized, meaning each component uses only the result of other component but not heavily dependent. One module can be updated by keep old module and change the old to a new one at the same time. “Amazon Cloud” provides the tools to deploy the codes instantly without interruption. The services should not be down because of the updates. The current documentation will include architecture, code description etc. so that any new developer can catch up with the work in a fair amount of time.

Portability:

The application will run in Android and iOS. The application is only suitable for mobile devices. Different versions of operating systems will require slight change in the source code.

User requirements:

1. As a user, I want the ability to register, Login and start using the application under 1 minute.
2. As a user, I want the application to securely save my Login information in the database so that my data remains secure.
3. As a user, I want the ability to change my Login information with necessary verification in case of any security events.
4. As a user, I want the application to count total calorie of food with accuracy of over 90% so that I can rely on the results.
5. As a health company user, I want more detailed information about my food so that, I can properly instruct my patient or user about their diet.
6. As a user, I want the application to use auto-calibration technique to estimate the quantity of the food so that, I do not have put finger or a coin.
7. As a paid user, I want the application to show more detailed stats and insights than a non-paid user.
8. As a user, I want the application Interfaces to be responsive and intuitive so that, I do not have to wait for anything.
9. As a user, I want to see all the food I have scanned so far so that, I do not have to re-scan any of the past items.
10. As a user, I want the ability to manually enter the names and quantity of each items so that I can get accurate estimation.
11. As a user, I would like small buttons at the bottom of every interface so that I can switch between interfaces easily.
12. As a user, I want my data to be safely stored in the cloud.
13. As a user, I would like the application to be secured so that It does not get used to exploit my phone.
14. As a user, I want the ability to enter my weight so that, I can keep track of my weight.
15. As a paid user, I want the application to give nutrition insights based on my scan history so that, I can fix my diet.
16. As a user, I want my past manual entry history.
17. As a user, I want to give feedbacks after the software has shown the total calories.
18. As a paid user, I want the application to suggest me new recipes like the foods in my history.
19. As an admin user, I want the ability to monitor and block any user misusing our application.
20. As a user, I want the User interface designs to be consistent.
21. As a paid user, I want more responsive and neat UI designs than unsubscribed users.
22. As a user, I want the software to correctly identify each item so that, I the total calorie estimation is accurate.
23. As a user, I want the application to assume the same density of each items over the size of the bowl.

Operational Concept:

When the user takes a photo, the application will measure the size of the bowl holding those items with auto-calibration technique. Convolutional neural network will detect the individual items in the food and use the measurement from auto-calibration to determine the quantity of each item. Nutritional table will provide with item calories which will be multiplied with quantity to get total calorie in the bowl. User data will remain safe in the cloud. Different user’s data will be restricted to themselves. The application will not be misused to gain personal value from using user’s data. The main modes of the system are- Auto-calibration, CNN, Nutritional Table.

User classes: 1. Unsubscribed user.

2. Subscribed User.

3. Health Companies.

Support Environment: The application will be supported in “Android” and “iOS” operating systems.

Operational Scenarios:

1. Registration:

Download from “App store” or “Play store” -> Application splash screen -> registration with email and password -> starts using the application.

1. Scanning an item (un-subscribed):

User Login -> System authenticates -> User takes a photo -> Photo goes to cloud -> Cloud computes the total calorie ->Shows the result to the user.

1. Scanning an item (Subscribed):

User Login -> System authenticates as paid user -> User takes a photo -> Photo goes to cloud -> Cloud computes the total calorie -> Shows the detailed result to the user.

1. Manual Entry:

User Login ->System authenticates -> Opens the Manual entry interface-> User enters each item, and quantity->cloud computes the total calorie-> Shows the result to the user.

1. History:

User Login ->System authenticates ->User opens the history tab-> Application fetches that user’s past item->Shows the list of items to the user.

1. Weight Log:

User Login ->System authenticates ->User opens the weight log tab->User Enters the weight->System logs the weight with date->Shows 7 last weight and dates.

1. New Recipes:

User Login ->System authenticates that it is a paid user ->User opens the recipes tab->system fetches recipes based on the past item->Shows the list of recipes to the user.