# A Conversational Agent for Nutritional Coaching: Report

CONSTANTINOS ROMANTZIS* and PROF. IOANNIS KATAKIS†, University of Nicosia, Cyprus

We present, as proof of concept, a Conversational Agent (Chatbot) that aims to provide personalized and adaptive nutritional coaching. As most diet-oriented Chatbots on the market deal primarily with basic calorie tracking, we have identified a need in the industry for a Chatbot that goes above and beyond simple tracking and provides a method for everyday people to set and achieve their goals as well as Medical Practitioners to provide timely and 24/7 available personal interventions to their patients by leveraging our CA as a tool. Through our initial Research and Development phase we have created the foundations of an agent that can be deployed on Instant Messaging platforms to offer Nutritional Coaching, backed by multiple Natural Language Processing, Natural Language Understanding and other Machine Learning models, in an intuitive manner that anyone with access to a smartphone can experience.

Additional Key Words and Phrases: datasets, neural networks, conversational agents, entity extraction, rasa, NLP, Natural Language Processing, NLU, Natural Language Understanding, NLU, Machine Learning

**ACM Reference Format:**
Constantinos Romantzis and Prof. Ioannis Katakis. 2023. A Conversational Agent for Nutritional Coaching: Report. 1, 1 (August 2023), 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

The project aims to create a Conversational Agent (CA) or Chatbot as they're widely called that will act as Nutritional Coach in order to provide an interactive framework for everyday people as well as medical professionals and their patients. Traditional diet plans rely on following outdated diet plans that are static and offer little to no flexibility for the user as any alterations and diversions beyond the simple choice between two meals requires an active intervention by the Medical Authority monitoring the person undertaking the diet which is difficult to achieve when managing multiple clients. Moreover, while diet plans are a relatively mainstream and well implemented method of nutritional coaching there appears to be an inclination towards an "one size fits all" mentality when treating patients. As a result, there does not appear to be a clear distinction between dieting for weight gain or weight loss and dieting to manage medical conditions which means that both groups could benefit by more personalisation. Out DietBot aims to fill the gaps that arise by the aforementioned methods by providing a fully customisable and personalisable agent that acts as a round-the-clock interactive bridge that connects patients and clients alike with their medical professionals. This is vital to achieve as obesity and diseases can often be managed relatively painlessly provided the professionals can provide timely and effective intervention; Our CA aims to provide them with the ability to automate these tasks without sacrificing the quality of said intervention.

## 2 RELATED WORK

Several studies have shown that CAs can be effective in helping users adhere to a strict and sensitive regiment [1- 2]. For example, a study by Gong et al developed a CA that provided type-2 diabetics with support in self-managing their condition [1] with the results being favorable, showing that it was successful in providing the patients with the required intervention and assistance. Additionally, it has been reported that CAs, despite not originally believed to be effective, proved to be efficient and effective in providing mental health support which could lead to regiment adherence including following specific diets [2].

In addition to literature regarding the efficacy of CAs, we also reviewed the various apps that seem to be dominating the market in regard to Macronutrient and Calorie tracking, such as MyFitnessPal, Lifesum and CalorieMama, and tried to identify finished products or apps providing these services via a CA. While we didn't identify any CA-based Nutritional Coaching or Calorie tracking apps available on the market we did manage to identify two papers that dealt with that concept with one being a diet tracking agent [3] and the other being a CA that provides primarily recipe recommendations [4].

Despite the promising results of previous studies, there is still a need for further research on the effectiveness of CAs acting as nutritional coaches. Specifically, there is a need to investigate the long-term impact of these agents on users' dietary behavior and health outcomes by incorporating them in a way that will allow both everyday people to use them to meet their goals but also Medical Professionals to utilize them as valuable tools that can help them provide timely and adequate intervention that will be available to their patients on a 24/7 basis. Additionally, there is a need to explore the design of a CA that can adapt to users' changing dietary needs and preferences dynamically.

## 3 DATA COLLECTION AND PREPROCESSING

As the CA will be dealing with both relatively healthy people looking to manage their weight and patients that need delicate management of their symptoms via dietary intervention it is extremely important that the datasets we use are not only truthful and grounded in reality but also allow the models behind the agent to work as expected in order to provide the users with accurate results. Overall, the data we dealt with were primarily nutritional databases and ingredient lists (sentences) taken directly from recipes.

*Student.
†Project Supervisor.

Authors' address: Constantinos Romantzis, romantzis.c1@live.unic.ac.cy; Prof. Ioannis Katakis, katakis.i@unic.ac.cy, University of Nicosia, Nicosia, Cyprus.

### 3.1 Nutritional Information Database

For the database that holds the nutritional information that the agent will provide to the user we tested a plethora of pre-curated datasets that were available on Kaggle and other dataset sharing websites. The final database is composed of the entirety of a dataset hosted on Kaggle that contains 8,789 unique entries as well additional items that were collected manually by us in order to assist the matching process. The database is currently incorporated in the CA in the form of a CSV file that gets loaded in the Matcher component as a dataset and is then iterated according to the user's selection. The initial dataset contains over 70 unique macronutrient and micronutrient values for each item meaning we have the ability to easily expand the CA's capabilities at any point. The additional entries we have made to aid the Matcher currently contain only the essential macronutrient values but this will be expanded in the future.

### 3.2 Entity Extractor Training Dataset

The Entity Extractor works by leveraging a model that takes a sentence as input and isolates the food item and its corresponding quantity. To achieve this, the model was initially trained on a dataset of approximately 25 sentences to gauge its effectiveness. Immediately we noticed its ability to isolate food items with an accuracy of 93% and grams with an accuracy of 87%. Its ability to recognise and extract other units of measurement was not reliable. A more extensive dataset of approximately 1,000 sentences with a total of 2,000 entries was curated over a period of fifty days and was annotated in the appropriate format using a Named Entity Recognition (NER) tagger. The process of tagging the entities was handled by using an online NER tagger and manually tagging each entity in the following format:

```
“u<Sentece>” , [ (<starting position of entity>,<ending
position of entity>, 'MEASUREMENT'), (<starting position
of entity>,<ending position of entity>, 'FOOD')
```

Even though the finished model boasted an impressive extraction accuracy of over 90% for almost all food items and most units of measurement there did appear to be a certain degree of overfitting and the size of the finished model was contributing to the overexertion of some hardware limitations. After reviewing the dataset, it was trimmed down to approximately 200 entries that retain a balance in regards to the units with only a minor sacrifice in accuracy and perform adequately for the current stage of the CA.

## 4 CONVERSATIONAL AGENT

### 4.1 Rasa Framework

From the beginning of the project we chose to use the Rasa Framework as the foundation and Conversational Manager of the CA as they offer a wide array of features that sports an extensive amount of updated Documentation as well as an active community from which we can benefit during the Research and Development (R&D) phases of this project. After a period of extensive experimentation we opted to use the current, as of the project's inception, version which was Rasa 3.x. This came at the expense of missing out on the visual interface provided by Rasa X which is a web based Visual Environment that handles both the training and testing of the CA that's exclusive to Rasa 2.x. Despite the obvious advantages of having Rasa X, we concluded that the Quality of Life improvements made in the overall framework as well as the syntactical simplifications that Rasa 3.x offers over Rasa 2.x far outweighed the ability to use Rasa X as Rasa offers, by default, the ability to test the CA in an interactive manner via the Command Line Interface (CLI) by using the “Rasa interactive” command and the CA was already expected to be hosted on Instant Messaging (IM) platforms. Starting the project is extremely straightforward as each new project comes with all the necessary scripts pre-configured in order to have a basis with all the standard functionalities. After that, the user is free to change everything to fit the intended use as well as to use custom scripts and components via the use of Custom Actions. The framework also comes with a preconfigured basic Pipeline which the user is able to change at any point in time and includes the ability to natively leverage the Duckling Extractor (see section 4.4).

### 4.2 Databases

The agent is connected to two databases that handle the storage of the information as well as the information retrieval and necessary aggregations that the other components are tasked with performing.

*4.2.1 Nutritional Database.* The Nutritional Database is a static database that is currently housed in a CSV file that contains rows that each correspond to an individual food and its respective Macronutrient and Micronutrient contents. It contains a total of 8,947 food items with up to 74 unique features, such as calories, protein, carbohydrates and fats, scaled to a uniform serving of 100g for solids and 100ml for liquids.

*4.2.2 User Data.* The User Data is contained in a dynamic database that is updated and maintained using SQL and specifically MySQL via a python script that is integrated into the CA's backend using a custom action. It contains the user's ID, date of log, food item, quantity, calories, protein, carbohydrates, fat and meal. The reason why this is a dynamic database is because the CA is able to both update the database every time the user wants to log what they have eaten and use it to retrieve the information when the user requests to see what they have had so far. The use of SQL means that the CA is able to perform aggregations on the data based on the date and the meal if specified.

### 4.3 Entity Extractor Model

The Entity Extractor is contained in a separate python script (.py) in the form of a function and works by receiving the user's sentence and passing it into the model which then handles the extraction of the food items and their respective quantities and returns them in a tuple. The model itself uses the framework provided by the spaCY module which is an open source library offering powerful Natural Language Processing capabilities while remaining relatively lightweight to use. In order for the agent to be able to use this the function is imported as a module into the Custom Actions script provided by default when initializing a Rasa project.

## 4.4 Duckling Temporal Entity Extractor

In order to support the agent when logging the users' data the Duckling Entity Extractor was selected to handle the extraction of the dates from the users' messages as it comes almost preconfigured with every Rasa project. In order for the Duckling Extractor to be used with Rasa it needs to be running, in parallel to the rasa servers, in a Docker container. After it's up and running the process of having Rasa recognise it is as simple as adding the required information into the Pipeline that's assembled in the configuration file provided by Rasa when creating a project.

## 4.5 Scheduler

To take further advantage of the CA's ability to extract and use dates we have created a Scheduler that causes the CA to message the user at predetermined time to remind them to log their food for the day. Similarly to the entity extractor (see 4.3), it is contained in a function in a separate python script and it is imported via the Custom Actions mechanism. It works by taking the user's desired time to be reminded at, in the form of Hour and Minutes (e.g "remind me every day at 21:15), the hours and minutes are then separated and multiplied by 3,600 and 60 respectively in order to be converted into seconds. The function then runs an infinite while loop that calculates the local time as per the loop and checks, using an IF statement, if the local time is before or after the reminder time. Depending on the current point in time it calculates the difference in seconds between the current time and the reminder time and sleeps for the duration of that period of time. To allow for the CA to run as expected while the scheduler is sleeping the function is run Asynchronously when triggered by the Custom Action. Once the sleeping time has passed the function triggers a webhook that forces the CA to initiate the conversation. This has been tested extensively by creating a mock server with Flask and using it to test the status of the webhook which always returns 200.

## 4.6 Entity Matcher

After the Entity Extractor extracts the food items and their quantities the tuple that is returned is immediately passed to the matcher. The method of matching itself received some of the bulk of the research process as we considered a lot of options. The initial plan was to use some method of Semantic matching with Elasticsearch being one of the frontrunners. Due to the nature of the CA as well as the fact that its stage is currently the "proof of concept" we determined that it might be excessive. We then considered replacing the matcher and the nutritional database with an API that provides the nutritional information directly; however, this was also deemed excessive for its current stage. Instead we opted for a more classic approach and decided to go with FuzzyWuzzy which is a database that handles string matching. In our case we installed the appropriate Levenshtein Distance package along with which enables it to use Levenshtein Distance to arrive as the closest match.

Levenshtein Distance works by calculating the number of edits that need to take place for two strings to match; the one that requires the least is considered the closest match.

It is calculated as follows:

$$\operatorname{lev}_{a,b}(i,j) = \begin{cases} \max(i,j) & \text{if } \min(i,j) = 0 \\ \min \begin{cases} \operatorname{lev}_{a,b}(i-1,j) + 1 \\ \operatorname{lev}_{a,b}(i,j-1) + 1 \\ \operatorname{lev}_{a,b}(i-1,j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise} \end{cases}$$

## 4.7 Assembled Agent

All of the external components are connected to the CA via the Custom Actions component that comes pre configured when creating the project (Actions.py) which allows the developers to create custom functions that can take advantage of Libraries and custom scripts like the proprietary components we have created in our case. These custom actions can then be called natively like all the default rasa actions. The processes that the CA follows can be scripted manually in the stories section of the backend (stories.yml) but a more efficient way to achieve that, which we followed, is to run the CA in interactive mode via the CLI which allows us to select how the CA will behave in a step by step fashion. Once that is finished it exports the session in the stories file which means that the next time the model is trained it will have that behavior scenario included in its training data. Lastly, multiple slots and forms were configured manually in order to be used by the custom actions that utilize the components introduced in the previous subsections.

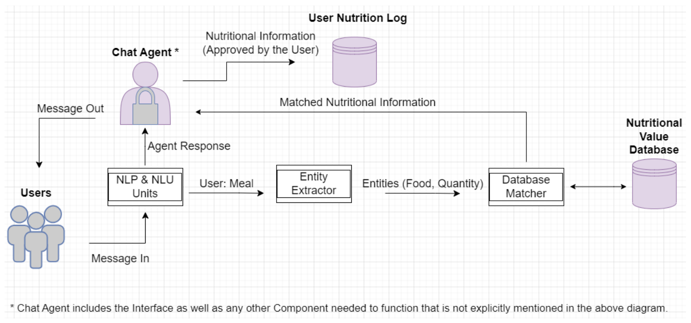
User Nutrition Log
Chat Agent *
Nutritional Information (Approved by the User)
Message Out
Matched Nutritional Information
Nutritional Value Database
Agent Response
Users
NLP & NLU Units
User: Meal
Entity Extractor
Entities (Food, Quantity)
Database Matcher
Message In
* Chat Agent includes the Interface as well as any other Component needed to function that is not explicitly mentioned in the above diagram.

Fig. 1. Connectivity between the user, CM and the individual components.

#### 4.7.1 *Initial Iteration.*

The first working version of the CA made use of four slots; Breakfast, Lunch, Dinner and Snack. These slots were filled sequentially by calling a custom action which would run and fill each slot's corresponding form before passing all the meals to the Nutritional Database in order to calculate the total calories and then to the User's Database in order for the food and quantity to be logged under each meal. This method, although perfectly functional, lacked the ability to add individual items that may or may not belong to a specific meal and the fact that you had to input them all at once proved to be a tedious process. Additionally, further attempts to introduce more slots would substantially increase the Agent's response time as the slots were filled mid-interaction while the user awaited the results.

*4.7.2 Version Presented.* The version that is currently demonstrated has kept the foundational functionalities provided in the previous version but demonstrates a significant overhaul in the way food is logged and slots are filled. The moment the interaction starts the user is presented with three buttons prompting them to select whether to log food, ask their current total calories or to Schedule a daily reminder. While the user is presented with these choices the CA automatically fills the User Id and Date slots in order for this information to be readily available if a logging or query takes place. Due to the ability to use buttons the matcher has also been updated. The matcher remains the same but the function that it utilizes had logic introduced that takes the matcher's match-confidence rating and then either returns the match if it's over 0.90 or presents the user with three buttons that each contain one of the three closest matches in descending order based on the matcher's confidence for each. After the selection is made the food is logged to the database along with the user's ID and the slots are reset in order for them to be filled again in the same manner when the user initiates the conversation again. Due to recently arisen technical difficulties the Duckling Extractor is not functioning properly and as such, for demonstration purposes, the CA has been reverted to an older version that does not feature some of the QoL improvements implemented recently as well as the custom action that causes the CA to retrieve the user's daily Calories and Macronutrients based on their ID and current Date.

*4.7.3 Version Currently in Development.* The most recent version, that is currently under debugging, has all the features mentioned in the previous section with the addition of an additional extraction model that extracts entities relating to the meals. This allows the user to specify a specific meal, if desired, which is logged along the food in the database and enables the user to request the total calories and macronutrients consumed during specific meals.

## 5 RESULTS

Technical setbacks aside, the agent was deployed on Meta Messenger on a testing basis and showed promising results. Due to the infrastructure provided by the IM platform, the agent was able to fully take advantage of the interface which enabled it to mimic a user-to-user interaction thus confirming our hypothesis that deploying a Nutritional Coach CA on IM platforms would provide users with an intuitive and familiar experience. Furthermore, the stable version (see 4.7.2) was recently showcased to an audience composed of high school students and their teachers at an Artificial Intelligence and Data Science conference, hosted by the Computer Science department of the University of Nicosia, with the demonstration being deemed a success. This further reinforces that this can be a really effective way to allow professionals to connect with their clients and patients and provide nutritional coaching in a manner that is both effective and adaptive.

## 6 CONCLUSION

During this project we have identified a potential gap in CA-leveraged Nutritional Coaching and specifically in the area where Nutritional Coaching intersects with proper Professional interventions either for simple weight management or for disease management via diet. We used the Rasa framework and designed a CA that uses custom components to perform entity extraction in order to recognise the food items and their quantities in the user's sentences and then matches them in an extensive list of foods and their macro and micronutrients with almost 9,000 entries. The resulting chatbot was deployed on Meta Messenger to test its ability to function properly via the interface and environment and it was then demonstrated in person to a small audience. This proof of concept acts as a validation of what we're trying to achieve. Moving forward we are aiming to improve its current performance but also introduce new features that will take further advantage of NLP and other ML models.

## 7 FUTURE WORK

In the same vein as the progress we have made so far, we are also working on additional features that we aim to have ready for implementation in the near future. These features cover a vast array of sections and offer multiple functionalities to the CA.

### 7.1 Image Classification Module

An image classification module is being worked on that will allow the user to get nutritional information on-the-go as well as to log their food directly by providing the CA with a picture of their meal. This will take advantage of the interface provided already by the IM platforms which will directly mimic the process of sharing images over chat in an everyday user-to-user interaction scenario.

### 7.2 Suggestions & General Ailment Guidelines

As this project aims to produce something more than a simple calorie counter we are actively working on adding coaching capabilities to the CA in the form of alternative food suggestions in order to give the users the ability to make informed decisions on the go. In line with this feature we are working on a module that will leverage a database that will consist of simple nutritional guidelines for each ailment identified by the medical professionals. It will work by allowing each medical professional to assign their patients a code that will correspond to the specific disease they're managing or their overall goal. These guidelines will be cross referenced before the agent makes nutritional suggestions to make sure the suggestions are appropriate for each user. For non-medical users the option to add a specific goal will be unlocked and the guidelines will be general weight management suggestions that will be decided, just like the medical guidelines above, by trained and qualified professionals.

### 7.3 Large Language Model Integration

As we have stated before, it is extremely vital that the nutritional information that the CA provides the user with is as accurate as possible in order to help them manage any delicate dietary demands they may have. As Large Language Models (LLMs), despite their impressive capabilities, are notorious for "Hallucinating" facts, at this point we have decided not to use their nutritional knowledge and instead rely on our curated nutritional information database. However, we are looking into the possibility of incorporating LLMs, like GPT-4, via the use of APIs to provide unit conversions in order to allow the use of a wider selection of units. As volume to weight conversions and vice versa rely on the density of each item, using LLMs to handle the initial unit conversion for each item is definitely an application that is worth looking into. If this produces encouraging results we will consider expanding the use of the LLMs.

## 8 REFERENCES

[1] Gong et al. 'My Diabetes Coach, a Mobile App–Based Interactive Conversational Agent to Support Type 2 Diabetes Self-Management: Randomized Effectiveness-Implementation Trial'. Journal of Medical Internet Research 22, no. 11 (5 November 2020): e20322.
https://doi.org/10.2196/20322.

[2] Sweeney, Colm, Courtney Potts, Edel Ennis, Raymond Bond, Maurice D. Mulvenna, Siobhan O'neill, Martin Malcolm, et al. 'Can Chatbots Help Support a Person's Mental Health? Perceptions and Views from Mental Healthcare Professionals and Experts'. ACM Transactions on Computing for Healthcare 2, no. 3 (31 July 2021): 1–15.
https://doi.org/10.1145/3453175.

[3] Patil, Dr Dipti, Surekha Iyer, Pooja Mehta, and Deesha Gavand. 'Dietbot - Diet Recommending Chatbot' 7, no. 11 (n.d.).

[4] Álvaro Samagaio, Henrique Lopes Cardos and David Ribeiro. Chatbot for food preferences modelling and recipe recommendation, n.d.