The Meal Finder

Logan McGloine
Pierce Hurd
Sven Kappeler

Problem

When trying to make a decision on where to eat it can be difficult to pick a specific restaurant to eat at, even if you know what you want.

This may be because you do not know what restaurants are in the area or because you aren't exactly sure which restaurant will best fit your preferences.

Our Solution

Our solution is to take information from the user and interpret it into a recommendation. Our approach uses a cognitive model to attempt to determine the users mental state and use that to match them with a restaurant.

Our Model

What is a cognitive Model?

A cognitive model recommender is a system that attempts to identify the users mental state and utilises that information to make an appropriate recommendation.

How is our system a cognitive Model?

Our model attempts to understand what a user is thinking by using the key words they input to the system. In this way we attempt to model there mental state and its from this model that we make our recommendations.

How it Works

- Our program takes in a sentence from the user and parses out important information
- The system then uses that information to generate a suggested category of food
- Then this is refined using key words from the user input
- A selection is made and presented to the user
- The user then has a chance to provide additional information to refine this suggestion

```
?- start.
What would you like to eat?:
|: i want steak
Selected Category: american
Do you want: rubytuesdays
If this is not what you want type a new statment
|: i want noodles
Do you want: pandaexpress?
If this is not what you want type a new statment
|: i want expensive food
Do you want: pandaexpress?
If this is not what you want type a new statment
|: done
true .
```

Belief revision

The beliefs of this program will constantly be revised as descriptors are added. This is because when you start with the initial set of preferences it may best fit one restaurant but once another preference is added a different restaurant will closer fit the preferences therefore changing the belief. This is something that may change many times before the final recommendation.

example:

if we start with of a initial descriptor of chicken our belief may be that they want Italian food, but then they add the descriptor of sweet, this would revise the belief to be chinese food.

Test Case

Initial Input: “I want cheap food”

Expected Return: “McDonalds”

Actual Return: mcdonalds

The result was expected because based on this limited information, just “cheap food”, since the Dish Item and Category were not specified we expect our result to be ‘the middle of the road option’ which we decided to be the “American” category.

Test Case

Next Input: “I want Chicken”

Expected Return: “KFC”

Actual Return: kfc

This is an example of our conceptual model doing a successful revision, it retains the information it was previously provided, “Cheap”, and adds a noun to our knowledge base allowing the system to narrow in on a specific restaurant. The restaurant we have in our knowledge base that fits the query the best is “KFC”

Test Case

Next Input: “I want Rice”

Expected Return: “KQ”

Actual Return: kq

This may seem confusing that with one extra Noun we are now being recommended a Chinese restaurant. This is working completely as intended however; if you recall the category was never specified and the only reason we were looking at American food in the first place was because that was just the default. In our system a combination of “Rice and Chicken” falls under the Chinese category, so now the system has more similarity with Chinese food than American food so it is now recommending Chinese restaurants, more specifically, cheap Chinese restaurants.

More Test Cases

Input: “I want expensive seafood”, Recommends: Snakebomb

Input : “I want expensive beef”, Recommends: Cheesecake Factory

Input: “I want cheep chicken”, Recommends KFC

Input: “I want average rice”, Recommends: KQ

Limitation

Recording Past Preferences:

Something our group was interested in implementing into our model was a way to record user's previous inputs

This would allow us to make a better first prediction

Top 3 Restaurants Return:

This isn't an issue currently but if there is a restaurant overlap (Same price, foods, category) one of the restaurants wouldn't be displayed

We tried implementing this but sometimes the two other restaurants were very inaccurate (occured during first few revisions when information was limited)

Code

Online code

Git Hub