Victoria Vynnychok

Foodie: NLP Food Recommendation Model

Abstract
Foodie is a food recommendation model that uses NLP based techniques in order to give user's food recommendations. The NLP recommendation model uses Word2Vec vectors as well as cosine similarity in order to recommend foods based on a user specified input of ingredients and keywords. The model can be integrated with a discord chatbot environment in the future to allow for users to interact with it and get recommendations in real time.

Problem Description
One universal experience for most of us is deciding what we should eat when we are hungry. In order to make this process easier, I decided to create a model that is able to output food recommendations for users based on their input of ingredients.

Data Description
I use the Food.com dataset, specifically the RAW_recipes csv file in this project which had about 230186 data entries. This data includes columns such as ingredients, description, and ratings. I specifically used the ingredient and name columns for training my model. I also used the description column for name entity recognition, to extract keywords such as "sweet" and to add them to a new data column for a recipe.
https://www.kaggle.com/datasets/shuyangli94/food-com-recipes-and-user-interactions?select=RAW_recipes.csv

Approach
First, I set up a virtual environment to download all needed dependencies for the model and bot. Then, I set up a python Discord bot using documentation. I tested this bot to make sure it connects to a discord server and responses to a user. This was the initial set up before working on the NLP portion.

Next, I found a dataset and created a Google Colab notebook to work in. I loaded the data from my dataset and preprocessed the data by removing special characters and commas. I then tokenized the ingredient data and vectorized it using Word2Vec. I trained a Word2Vec model on the dataset using the ingredient and name columns and made sure the vocab list of the model was correct by printing the results. The next step was to create a function called recipe_vector that would check whether the tokenized and cleaned user input contained words that existed in the foodie Word2Vec model's word list, and to get the average of the vectors of those valid words. I compared the user's input vector with the recipe vectors of every single recipe in the dataset and

got the recipes with the highest similarities to the users' input. These were the final results that the user got for one example:

User input:

```
input_vector = recipe_vector(['cheese', 'bread', 'sugar', 'squash'], model)
```

Top cosine similarities:

```
print(similarities[top_indices])
```

```
[0.7775472  0.75041986 0.72733164 0.7012359  0.698444  ]
```

Top recipe names and ingredient lists:

```
for recipe in top_recipes:
    print(recipe)

for ing in top_recipes_ing:
    print(ing)
```

```
cheeze bread
caramelized cheese covered grilled cheese sandwich
pear and cheese toast
kimke s grilled cheese
extra special grilled cheese sandwich
['bread', 'cheese']
['butter', 'bread', 'cheese']
['pear', 'cheese', 'bread', 'butter']
['bread', 'cheese', 'butter', 'season salt']
['bread', 'cheese', 'butter', 'jelly']
```

Additionally, I extracted keywords from the description column of the dataset for each recipe using name entity recognition, and created a new column for keywords. This is so the data can be further processed to get the results the user desires.

Experiments and Error Analysis

There were many attempts tried at creating both the recommendation model and discord bot.

In relation to the dataset, my first approach involved using SPARQL in order to query wikidata for food related data based on user input. I had trouble in my approach to querying the database for the exact data I wanted which involved all food names and their names and extracting that data. Therefore, after multiple attempts, I decided to find a dataset online from Kaggle instead. I found the Food.com recipe dataset which I determined to be sufficient due to the numerous amounts of tags and ingredients based data for each recipe name.

In relation to the NLP model, the first model did not recommend recipes that included the same ones that the user inputted. After doing some code and error analysis with multiple print statements, I found that the tokens being inputted into the Word2Vec model were characters instead of words that were composed of full ingredients. After removing tokenizing my data, the next error was the inclusion of commas and other special characters as tokens. This required further token and data cleaning, but afterwards I saw large improvements in the accuracy of my model which was measured by my personal satisfaction of the results and how high the cosine similarly results were. Additionally, I found an issue where certain words that had the same meaning such as ‘cherry’ and ‘cherries’ provided different outputs due to them being considered different inputs. Including both the singular and plural version of an ingredient would create the best results catered towards important words like apples and cherries.

In relation to the discord bot, there were multiple issues related to installing packages when integrating the model from the foodie.ipynb file to the chatbot.py file. I downloaded the model from my google colab notebook, naming the file foodie.model. I also downloaded the cleaned and tokenized dataset recipe and training X and y data with pickle. The problem I encountered was installing gensim in my Visual Studio Code environment as well as Word2Vec which was needed to load my model that I downloaded to foodie.model. After spending numerous hours trying to alter my setup, I determined that while both the model and the demo discord bot work separately, their integration is currently unsupported due to great difficulty in installing gensim and Word2Vec in my current system environment.

Conclusion

The Word2Vec model takes user input and recommends the recipe that is closest to what you want to eat. This model works on a simple basis but can be improved in the future in many ways such as by recommending users the highest rated recipe name, providing links to the recipes, and letting the user ask for food related to the meal time of day such as ‘lunch’ or ‘dinner’. It can also do better to handle user ambiguities by improving to process the data. In order to test the accuracy of the model, there would have to be user testing to determine how close the results are to the actual given user inputs. It is not as simple as taking an accuracy analysis of the model, because the model uses Word2Vec vectors, and then compares those vectors to the user inputted vector of ingredients using cosine similarity. While the top cosine similarity results can potentially be used to measure the accuracy of the model, I believe it needs a more complex and human measurement of accuracy due to user’s not always wanting a recipe with only the exact ingredients they have included in their input. Typically, variation is wanted. With additional processing of keywords, the accuracy metric again is determined to be human opinion itself. For the cosine similarities, I got result of about 70% similarity for most of the entries I created and I was satisfied with the results because they matched the input that I wanted. Therefore, foodie is a

good basis model for food recommendation based on ingredient lists, but can be improved with further processing.

References

Discord Documentation to Set up Python Based Bot: https://discordpy.readthedocs.io/en/latest/