Recipe Extractor

Ke Zhou, Pragya Srivastava, Sheldon Sebastian

Introduction

The goal of this project is to build a standardized (machine readable) recipe knowledge database. This will be done by extracting, classifying, and grouping information found in food recipes like actions to perform on ingredients, utensils used and identifying the ingredients and quantities. In this project, we aim to reduce the preparation steps of a recipe to ACTION, INGREDIENT, UTENSIL groups. For instance, the sentence "Chop the tomatoes and place them in a large skillet" should be reduced to following two groups: CHOP(TOMATOES); PLACE(TOMATOES) (SKILLET).

Background

Food recipes have a standard template and converting recipes into a machine-understandable knowledge base has a wide variety of applications. The knowledgebase can be used by home assistants like Alexa or Google Home to guide users in cooking.

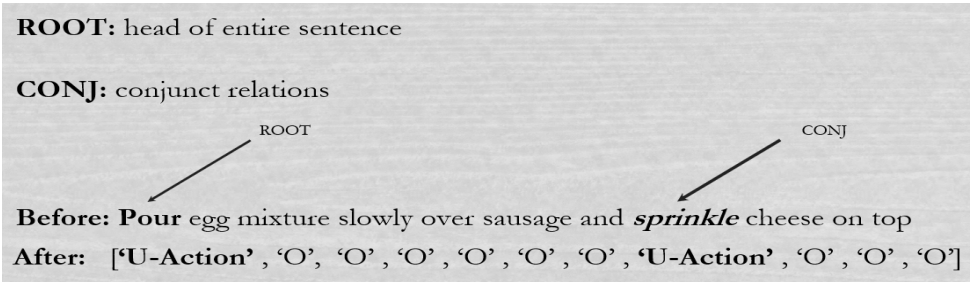
We read a couple of papers and research work done by others in this field and were most influenced by the following in formulating our approach - A Statistical Machine Learning Approach to Generating Graph Structures from Food Recipes, Information Extraction From Recipes, Extracting Structured Data From Recipes Using Conditional Random Fields.

Initially we planned to use a dataset available on Kaggle - 'Food.com Recipes and Interactions'. It contains about 200K recipes with multiple columns such as Recipe Name, Nutrition, Steps and Ingredients etc. But it contained unnecessary elements like 'Notes' at the beginning of each recipe, some steps were not in proper sequence so we decided to go ahead and scrape the website, Food.com, ourselves, collecting only data that we needed and in a format that suited our purposes.

Scope

- ***Annotating the data*** – We use the BILOU tagging method to annotate the data on DataTurks. The BILOU method works better that IOB for our case because it accounts for multi token entities like olive oil. Given below is an example of how the BILOU tagging works –

- ***Train-Test Split*** -We then split the data into train and test. It was a total of 264 sentences, out of which 212 went to the training/validation set and 52 went to the test set.
- ***Baseline Model – Dependency Parsing –*** The recipe dataset mainly contained imperative sentences therefore the context free parser did not work well for us, which is why we implemented the dependency parser. The main advantage of the dependency parser over the traditional parser is that it can extract key information from a sentence because it takes the entire sentence into account. Using the dependency parser, we extracted 'root words' which are the head/start of the entire sentence and their 'conjunct relations' and tagged both as U-Action words and the remaining words in the sentence as 'Other'. The example provided below makes this idea clearer –



- ***Conditional Random Fields –*** To improve the results we achieved from the dependency parser, we made use of conditional random fields.
- ***Ingredient-Action Pairs –*** As an extension of the above exercise we also identified ingredient action pairs likely to occur together. For this, we made use of association rules and calculated the likelihood of the various combinations of ingredients and actions occurring in the recipes.

Outcomes

- The BILOU annotation method gave us the following distribution of tags. In our study we were primarily interested in the action words, followed by ingredients.

```
1  data["tag"].value_counts()
```

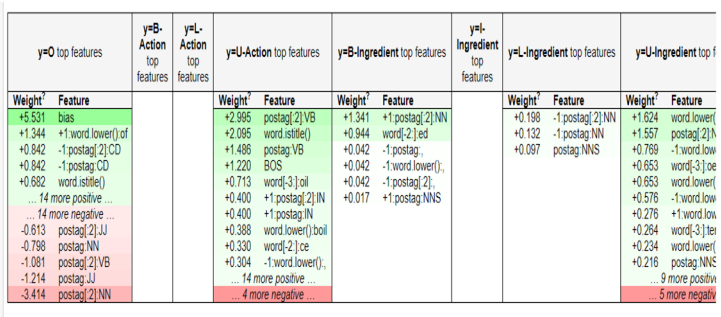
```
O                2498
U-Action          366
U-Ingredient      241
B-Ingredient       70
L-Ingredient       70
B-Utensil          60
U-Utensil          59
L-Utensil          59
I-Utensil          15
I-Ingredient        3
B-Action            2
L-Action            2
Name: tag, dtype: int64
```

- Given below are the results that we have from the dependency parsing technique. This method proved to be a useful baseline model but did not result in very high accuracy scores which is why we made use of conditional random fields.

	precision	recall	f1-score	support
U-Action	0.523	0.695	0.597	82
micro avg	0.523	0.695	0.597	82
macro avg	0.523	0.695	0.597	82
weighted avg	0.523	0.695	0.597	82

- The results from the unregularized CRF (results available in the appendix) model and feature importance table for this model (available in appendix) showed that the model was assigning higher weights to certain words and features which it shouldn't, also the f-score was high which raised suspicion of the model overfitting the data. We then added regularization to the model, and the results improved as visible from the feature importance table (available in appendix) and the precision recall scores, the model no longer overfits–

	precision	recall	f1-score	support
U-Action	0.877	0.695	0.776	82
U-Utensil	1.000	0.118	0.211	17
B-Utensil	0.500	0.077	0.133	13
L-Utensil	0.500	0.077	0.133	13
U-Ingredient	0.681	0.711	0.696	45
B-Ingredient	0.304	0.583	0.400	12
L-Ingredient	0.304	0.583	0.400	12
I-Utensil	0.000	0.000	0.000	3
I-Ingredient	0.000	0.000	0.000	0
B-Action	0.000	0.000	0.000	0
L-Action	0.000	0.000	0.000	0
micro avg	0.652	0.543	0.593	197
macro avg	0.379	0.259	0.250	197
weighted avg	0.710	0.543	0.566	197



y=O top features		y=B-Action top features	y=L-Action top features	y=U-Action top features		y=B-Ingredient top features		y=I-Ingredient top features	y=L-Ingredient top features		y=U-Ingredient top f	
Weight	Feature			Weight	Feature	Weight	Feature		Weight	Feature	Weight	Feature
+5.531	bias			+2.995	postag[:2]:VB	+1.341	+1:postag[:2]:NN		+0.198	-1:postag[:2]:NN	+1.624	word.lower(
+1.344	+1:word.lower():of			+2.095	word.istitle()	+0.944	word[-2:]:ed		+0.132	-1:postag:NN	+1.557	postag[:2]:N
+0.842	-1:postag[:2]:CD			+1.486	postag:VB	+0.042	-1:postag:,		+0.097	postag:NNS	+0.769	-1:word.low
+0.842	-1:postag:CD			+1.220	BOS	+0.042	-1:word.lower():,				+0.653	word[-3:]:oe
+0.682	word.istitle()			+0.713	word[-3:]:oil	+0.042	-1:postag[:2]:,				+0.653	word.lower(
… 14 more positive …				+0.400	+1:postag[:2]:IN	+0.017	+1:postag:NNS				+0.576	-1:word.low
… 14 more negative …				+0.400	+1:postag:IN						+0.276	+1:word.low
-0.613	postag[:2]:JJ			+0.388	word.lower():boil						+0.264	word[-3:]:te
-0.798	postag:NN			+0.330	word[-2:]:ce						+0.234	word.lower(
-1.081	postag[:2]:VB			+0.304	-1:word.lower():,						+0.216	postag:NNS
-1.214	postag:JJ			… 14 more positive …							… 9 more positiv	
-3.414	postag[:2]:NN			… 4 more negative …							… 5 more negativ	

- We then tested the above model on untagged data and got the below results, which shows that the model seems to be working well.

	precision	recall	f1-score	support
U-Action	0.870	0.777	0.821	319
U-Utensil	1.000	0.286	0.444	63
B-Utensil	0.636	0.237	0.346	59
L-Utensil	0.636	0.233	0.341	60
U-Ingredient	0.711	0.581	0.640	258
B-Ingredient	0.245	0.529	0.335	51
L-Ingredient	0.236	0.520	0.325	50
I-Utensil	0.000	0.000	0.000	17
I-Ingredient	0.000	0.000	0.000	4
B-Action	0.000	0.000	0.000	1
L-Action	0.000	0.000	0.000	0
micro avg	0.639	0.563	0.599	882
macro avg	0.394	0.288	0.296	882
weighted avg	0.708	0.563	0.600	882

Challenges

- While parsing the raw recipes, we found that the imperative sentence structure, found in recipes, is the main obstacle to overcome.
- Tagging recipes manually is a time-consuming process which led us to reduce the size of our train and test data

Future Work

The long-term goal of this project is developing a high-performance recipe extractor. It aims to identify key elements from any recipe of choice with high accuracy. As an extension of this exercise, we also extracted ingredient action pairs which commonly occur together – for instance eggs are usually whisked or beaten in recipes. For achieving this we first extracted ingredient (noun) – action (verb) pairs occurring in all recipes and then using association rules, calculated confidence for each pair of ingredients and actions. This is a probabilistic method and assigned probability values to ingredients and actions which are more likely to occur. Some examples of the outputs are attached in the Appendix.

Appendix

- Results of the unregularized CRF Model

y=O top features		y=B-Action top features		y=L-Action top features		y=U-Action top features		y=B-Ingredient top features	
Weight[?]	Feature	Weight[?]	Feature	Weight[?]	Feature	Weight[?]	Feature	Weight[?]	Feature
+2.215	bias	+0.583	-1:word.lower():and	+0.458	-1:word.lower():brown	+1.811	postag[:2]:VB	+1.209	+1:postag[:2]:NN
+1.159	+1:word.lower():of	+0.546	-1:postag[:2]:CC	+0.454	word.lower():lightly	+1.430	word.istitle()	+0.679	word[-2:]:ed
+0.878	-1:postag:CD	+0.546	-1:postag:CC	+0.447	word[-3:]:tly	+1.262	postag:VB	+0.667	+1:word.lower():cream
+0.878	-1:postag[:2]:CD	+0.419	+1:word.lower():lightly	+0.432	word[-2:]:ly	+1.070	BOS	+0.655	+1:postag:NNS
+0.858	postag[:2]:IN	+0.399	+1:word.lower():up	+0.417	-1:word.lower():roll	+1.028	word.lower():boil	+0.537	-1:word.lower():with
+0.858	postag:IN	+0.395	word[-3:]:oll	+0.408	postag:RB	+0.917	word[-3:]:oil	+0.534	+1:word.lower():sugar
+0.830	EOS	+0.395	word.lower():roll	+0.408	postag[:2]:RB	+0.795	word.lower():cover	+0.502	word[-3:]:und
… 1997 more positive …		+0.394	word.lower():brown	+0.406	word.lower():up	+0.760	-1:word.lower():before	+0.502	word.lower():ground
… 357 more negative …		+0.393	word[-3:]:own	+0.406	word[-3:]:up	+0.731	postag:VBG	+0.500	+1:postag:NN
-0.827	-1:word.lower():by	+0.393	word[-2:]:wn	+0.400	postag:RP	+0.714	word[-3:]:eat	*… 183 more positive …*	
-1.105	postag:NN	*… 7 more positive …*		*… 12 more positive …*		*… 344 more positive …*		*… 24 more negative …*	
-1.800	postag[:2]:NN	*… 11 more negative …*		*… 9 more negative …*		*… 90 more negative …*		-0.503	-1:postag[:2]:NN

	precision	recall	f1-score	support
U-Action	0.889	0.878	0.883	82
U-Utensil	1.000	0.353	0.522	17
B-Utensil	1.000	0.231	0.375	13
L-Utensil	1.000	0.308	0.471	13
U-Ingredient	0.756	0.756	0.756	45
B-Ingredient	0.538	0.583	0.560	12
L-Ingredient	0.538	0.583	0.560	12
I-Utensil	1.000	0.333	0.500	3
I-Ingredient	0.000	0.000	0.000	0
B-Action	0.000	0.000	0.000	0
L-Action	0.000	0.000	0.000	0
micro avg	0.807	0.680	0.738	197
macro avg	0.611	0.366	0.421	197
weighted avg	0.842	0.680	0.717	197

- Github Link - https://github.com/sheldonsebastian/NLP-Project
- Results of noun-verb pair extraction, i.e. recipe and action pairs:

Ingredient	Action	Confidence Percentage
onion	saute	38.0
onion	add	25.0
onion	brown	25.0
onion	chop	12.0
sugar	burn	17.0
sugar	cream	17.0
sugar	mix	17.0
sugar	spread	17.0
sugar	add	17.0