Word2Mouth: A Top Dish Recommender System

Abhirup Mukherjee
amukherjee@umass.edu

Mithra Muthukrishnan
mmuthukrishnan@umass.edu

Aditya Narula
anarula@umass.edu

Aishwarya Turuvekere
aturuvekere@umass.edu

1 Problem statement

Yelp provides a convenient way to discover restaurants with top reviews by users. However, besides these reviews, most customers are also interested in information about the businesses themselves, such as menus for restaurants and the restaurant's most popular dishes in order to make a decision regarding not only what to order at the establishment, but also to choose to visit the restaurant at all. Most restaurants on Yelp don't have menu items or recommended dishes and this information is buried in large amounts of user review data. This entails more work for users, including, but not limited to, reading and parsing a sufficient amount of reviews by other users to get an idea of what the popular dishes are at an establishment and what these dishes look like.

We propose a system to automatically detect and extract dishes from reviews and create a list of popular dishes for each restaurant. Our solution consists of a pipeline split into two overarching modules. The first module is a Named Entity Recognition system that trains itself on a small sample of annotated user review data and then extracts the food items in each review automatically, using food as a category of entity. The second module functions as a Sentiment Analyzer that assigns a sentiment value to the review in which a food item is mentioned. Using these two modules in conjunction, we hope to first extract all unique food items mentioned in a set of user reviews and the compile a list of top dishes using the aggregated sentiment scores of the reviews in which each of those food items are mentioned. We aim for the proposed system to be independent of the availability of menus, which are often not available or difficult to source. The only requirement would be a large enough corpus of reviews to allow our system to accurately extract the top dishes information. This system would be relevant to all types of restaurant goers, from the casual diners who are new to the city or are exploring the area for the first time, to avid food connoisseurs trying out the food at a well rated restaurant for the first time. This would also be valuable to restaurant owners to keep track of the most sold dishes, without having to sift through thousands of reviews.

2 Dataset

Our original plan was to utilize a small subset of the yelp dataset, available through the yelp dataset challenge, to train both our named entity recognition model and our sentiment analyzer which would then work in conjunction to identify the dishes from the reviews of a restaurant and then rank them to create a top dishes list. However, we ran into some trouble with this approach due to the fact that the yelp dataset is slightly outdated, and only contains data for very specific restaurants from only around a dozen cities from around the world. The biggest city in the dataset was Los Angeles and the dataset had several consistency and cleaning issues, alongside the fact that we needed restaurants that already had a list of top dishes that we could validate our final results against. Due to the failure of this approach, our second approach was to scrape the yelp website ourselves. The data requirements of our system involve data to train the two modules of our pipeline and data to validate the results of each module individually. This meant that we needed to collect one dataset containing reviews from restaurants with top dishes and menus available on the yelp website, and a second dataset containing all the dishes on the menu for the restaurants in our reviews dataset. This involved using Helena, a chrome extension that allows for automation of repetitive interactions with well-structured web-

pages, ie, a scraper. Using the Helena scraper, we targeted some of the top restaurants in four major cities: San Francisco, New York City, Chicago and Seattle. We collected approximately 500 - 1000 reviews for 5 restaurants in each of the 4 cities for a cumulative of 2905 rows. We also collected a menus dataset containing 1398 rows. The review dataset at the time of collection consisted of 5 columns: restaurant_name: name of the restaurant, user_name: name of the user giving review, star: star rating given by user (from 1 to 5), date: date of review, text: the review text itself. The menus dataset at the time of collection consisted of 3 columns: restaurant_name: name of the restaurant, menu_item: name of the dish, is_popular: indicator variable that is set to 1 if the dish is popular and 0 if dish is not. To standardize the dataset before we started cleaning, tokenizing and annotating it, we replaced the restaurant_name columns in both datasets with a business_id column that contained a numerical identifier for each restaurant. A master business dataset contained a central mapping of each of the twenty restaurants to a unique numerical identifier. We also created a review_id column to uniquely identify each review in our dataset.

business_id	name	city	state	stars	categories
1111	Delmonico's	New York	NY	4	¬†Steakhouses,¬†American (Traditional)
1112	Mad Dog & E	New York	NY	3.5	Mexican, Bars
1113	Katz's Delica	New York	NY	4	Delis, Sandwiches
1114	Luke's Lobste	New York	NY	4.5	Seafood
1115	Totto Ramer	New York	NY	4	Ramen
1116	Tartine Bake	San Francisc	CA	4	Bakeries, Cafes, Desserts
1117	San Tung	San Francisc	CA	4	Chinese,¬†Chicken Wings,¬†Noodles
1118	DOSA	San Francisc	CA	4	Indian
1119	Fog Harbor F	San Francisc	CA	4.5	Seafood
1120	Gary Danko	San Francisc	CA	4.5	American (New),¬†French,¬†Wine Bars
1121	Pike Place Ch	Seattle	WA	4.5	Seafood, Soup
1122	Metropolitar	Seattle	WA	4	American (Traditional), Steakhouses
1123	Quinn's	Seattle	WA	4	American (New), Pubs
1124	Local 360	Seattle	WA	4	American (New), Cocktail Bars
1125	Toulouse Pet	Seattle	WA	4	Cajun/Creole, Bars
1126	Girl & the Gc	Chicago	IL	4.5	American (New),¬†Bakeries,¬†Coffee & Tea
1127	Wildberry Pa	Chicago	IL	4.5	Breakfast & Brunch,¬†American (New)
1128	The Purple P	Chicago	IL	4	Tapas/Small Plates,¬†Mediterranean,¬†Breakfast & Brunch
1129	Smoque BBC	Chicago	IL	4.5	Barbeque
1130	Alinea	Chicago	IL	4.5	American (New),¬†Modern European¬†

Figure 1: Snapshot of the business dataset

business_id	menu_items	is_popular
1111	East Coast Oysters	0
1111	West Coast Oysters	0
1111	Crab Cocktail	0
1111	Maine Lobster	0
1111	Shrimp Cocktail	0
1111	Shellfish Chateau	0
1111	Cocktail Collection	0

Figure 2: Snapshot of the menus dataset

review_id	user_name	business_id	stars	date	text
1	Gauri H.	1111	5	11/3/18	oldest stakehouses in new york .great quality of stake. very flavorf
2	Jason M.	1111	5	10/19/18	wow! incredible meal indeed.i came her for a friday lunch with m
3	Marisa S.	1111	5	10/11/18	went here on a date because my date saw it on john wick. when w
4	Chris L.	1111	5	10/7/18	i don't know what to say but simply one of the best dinners and st
5	Dolonchapa (	1111	5	8/26/18	we came here for restaurant week. they are one of the few places

Figure 3: Snapshot of the reviews dataset

Once our scraped datasets were ready, the next logical steps involved eliminating duplicate reviews, cleaning the review text, and tokenizing the review text. While eliminating duplicate reviews was fairly simple, we found that cleaning real world reviews scraped from the internet is a difficult task. Problems with the data included irregular spellings, non english characters, special symbols, unicode character errors, irregular grammar and punctuations amongst others. To deal with these issues in a standardized manner we decided to remove all numbers, special characters, non english characters, unicode character errors, repetitive punctuation symbols and whitespaces before using the data to train our modules. We also tokenised the review text using whitespace as the delimiter prior to training the sentiment analyzer.

3 Annotation

In order to train an NER model, we needed to annotate the data and separate it into train and test. From the 2905 reviews scraped from the 20 restaurants, we have initially annotated about 100 of these reviews with a total of 971 mentions. We have split this into an 80 - 20 train and test set.
We are using an online annotation tool, *TagTog*, to annotate our reviews for the named entity recognition task. Each person in the team started with a set of 25 randomly picked reviews from a restaurant. The annotation tool allows the user to create his own entities and tag single or multiple words as a mention. To ensure the quality of the annotations, each batch of 25 reviews is annotated by two annotators. The results are combined to create a gold annotated set. For the sentiment analysis part of the project, we initially annotated 100 sentences to use as our train - test set. We used the labels positive, neutral, and negative to denote the sentiment of the reviews for the dish mentioned in the sentence.

3.1 Annotator Agreement

As with any annotation task, errors will occur. We evaluate our precision and accuracy by taking a 100 mentions and comparing the matches made by annotators. 90% of the annotations were exact matches. In 6% of the cases, the annotators disagreed whether the item mentioned in the review was a menu item. In 3% of the cases, the annotators disagreed whether to annotate phrases worded differently from the menu as a menu item and

whether ingredients should be tagged as dishes. Since our annotator tool does not merge conflicting annotations, these conflicts were merged on a case by case basis.
For the sentiment analysis task, only 4% of the labels given to sentences were different. This typically occurred when one annotator felt that the sentence was positive and another felt that the sentence was neutral.

4 Baselines

We consider the following baseline for the named entity recognition task - since there are a finite number of dishes, we can use a food dictionary, i.e. a large collection of dishes, ingredients and related words to tag the dishes that appear in the menu.
We are using the Diner's Dictionary, which contains over 2400 gastronomical words and phrases. The dictionary includes everything from common fruits and vegetables to foreign dishes like gado-gado, nasi goreng, and satay to name a few. The dictionary covers foreign cuisines like Thai, Korean, and Vietnamese. Compared to a few other dictionaries, we saw that this focuses on the dishes and not too much on the ingredients, which fits our task well. The entries were scraped to create a list which was then normalized. The reviews were normalized and tokenized, and we looked for words which appeared in the dictionary, calling them the mentions or entities. A simple assumption was made: if identified entities appeared next to each other, they would be treated as a single dish.
We noticed that the reviews which included the dishes often had simple inflections - "if i had more money, i would be awash in coconut cream pies, lemon cream tarts, and morning buns with a hint of orange." is a sentence from a review. Using absolute match, the baseline would not pick up dishes like "coconut cream pie", because the exact word pie does not appear in the sentence. To address this problem, we included a stemming step in our baseline. We are presently using the snowball stemmer from the NLTK package. We ran this baseline over our dataset mentioned in the previous sections and saw the following results -

Table 1: Precision-Recall of Baseline

	Baseline (with stemming)	Baseline (without stemming)
Precision	11%	6.85%
Recall	25%	10%

Even though our original idea was to include the stemmer to improve performance of the baseline, the above table shows that the baseline with the stemming included actually performs worse than the other one.

5 Error analysis

Let us look at the following examples where the baseline fails -

1. "the chocolate chip walnut cookie is big but very flat and super crispy.", here the expected menu item was chocolate chip walnut cookie which was only picked up by the baseline without stemmer. The stemmer often converts words like chocolate to chocol and cookie to cooki which arent found in the food dictionary.

2. "but i have not had a morning bun , cinnamon bun , breakfast roll that was as good as their morning buns .", here the menu item morning bun has a noun used in the name, which cannot be identified by either baseline.

3. "we also tried the idiazabal & membrillo sandwich and different loaves of bread ancient grain, porridge and walnut." - This menu item fails our simple assumption of words which are part of the same dish name are separated by one character. Usage of conjunctions and symbols like and and are not picked up by the baseline. In the example, menu item ancient grain bread is mentioned, but not in the same format. The reviewer instead says loaves of bread ancient grain, which is also not picked up by the either variations of the baseline.

4. "pain au jambon ham and cheese croissant excellent breakfast choice that was actually pretty big and filled me up." - This example is similar to the one above as it uses the word and in the dish name. Similarly, even though french words pain and jambon is found in the dictionary, the connecting word au is not.

The following is an example of an output from the NER model built with spaCy (explained in the **Approach**):
The gold standard provides the true annotations for the review. The number next to each food item

indicates how many times the item appeared in the review.
Gold Standard: 'milk braised pork shoulder': 1, 'octopus': 1, 'fingerling potatoes': 1, 'salsa verde': 1
Document: "finally tried this place and it was very very good. i did my best to finish the milk braised pork shoulder and octopus with fingerling potatoes and salsa verde. so delicious, no regrets!!"
Predictions milk braised pork shoulder
Predictions octopus with
Predictions potatoes
Predictions salsa verde

The spaCy model here is able to pick up words like "milk braised pork shoulder". This is something that the baseline model would have trouble picking up because the manner in which the pork shoulder is cooked is specific to the restaurant. However, there is a drawback in the evaluation metric used on the test data in the NER model. Currently, the word "potatoes" is seen as incorrect food, however the model should get credit for picking up potatoes even if it is without fingerling. We will have to implement different evaluation metrics in the future to have a more accurate test precision score. Though we have not implemented an bidirectional LSTM yet, we expect it to perform better than the baseline because the LSTM should learn both how entities look and where in the corpus entities are found.

6 Approach

Our system to automatically detect and extract dishes from reviews and create a list of popular dishes for individual restaurants consists of a pipeline split into two overarching modules, as seen in Figure 4. The first module is a Named Entity Recognition system that trains itself on a small sample of annotated user review data and then extracts the food items in each review automatically, using food as a category of entity. The second module functions as a Sentiment Analyzer that assigns a sentiment value to the review in which a food item is mentioned. Using these two modules, we hope to first extract all unique food items mentioned in a set of positive user reviews and the compile a list of top dishes using the aggregated sentiment scores of the reviews in which each of those food items are mentioned.

Both are modules' core function can be implemented using two approaches. For our progress report, we have implemented the Named Entity Recognition model using spaCy and we have implemented the sentiment analyzer using Logistic Regression. Upon completion of the project, we expect to have the other approach of our pipeline ready. The spaCy approach involves training the NER model using training data annotated in the format explained below. Once the NER model is trained, we use it to extract food entities from unannotated reviews. This gives us a list of food/-menu items for the respective restaurant who reviews we are identifying the top dishes for. A similarity metric is applied to the list to eliminate redundant menu items. The final list is then used with the sentiment model to determine the top dishes. This procedure allows us to bypass the requirement of needing yelp to store the menus for the respective restaurant we are identifying the top dishes for.

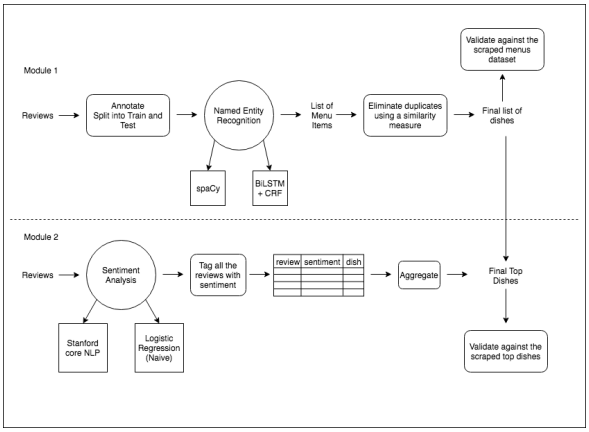

Figure 4: The System Pipeline

6.1 spaCy

We want to generate an a list of food items that are restaurant specific using spaCy. In order to achieves this, we must first build a named entity recognition (NER) model. Currently, NER models in existence do not identifiy food entities. We used spaCy to train an NER model based on the annotations outlined above. We started by manually annotating 100 reviews from 20 restaurants. spaCy requires the training data to be in a specific format. The following is an example of the format that spaCy expects as an input for the given sentence: "the french onion soup was good. steak was great. asparagus with bernaise was good. mashed potatoes were great. ceasar

salad was great. creme brulee was amazing." spaCy input:
('the french onion soup was good. steak was great. asparagus with bernaise was good. mashed potatoes were great.', entities : [(4,21,'FOOD'), (32,37,'FOOD'), (44,67,'FOOD'), (79,94,'FOOD')]) Thus, for each food item annotated in the review, an offset entity tuple is created with the form (start, end, label).
Internally, spaCy converts it to the BILOU format of annotating sentences. Once the annotations created with *TagTog* were converted into the offset entity tuples, we used an 80-20 split on the data and trained our model. The model goes through ten iterations, and updates each time based on the loss function provided by spaCy. For evaluating the test data, we could not use the built in "evaluate()" function or "scorer()" function because of inconsistencies in the expected parameter types. We wrote our own scorer function which calculates the precision, recall, and f1 scores. In order to calculate these metrics, we first had to find the true positives (TP), false positive (FP), and false negatives (FN). For our model, a true positive is defined as the entities which are predicted by the model and found in the gold standard. A false positive is defined as entities which are predicted by the model, but not found in the gold standard. A false negative are entities which are not predicted by the model, but do exist in the gold standard. Using these definitions, we calculate precision and recall with the following equations:

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$f1\ score = 2 * \frac{precision * recall}{precision + recall}$$

We trained our model using an increasing number of reviews as the training set, and plotted these metrics. As the number of reviews in the training set increase, the precision, recall, and f1 scores increase. Although it seems that there is a peak at 50 reviews, we will try to increase our training set further and see if we can increase the precision past this peak. Additionally, we plotted precision versus recall which is shown in the Figure 3. As precision increases, so does recall.

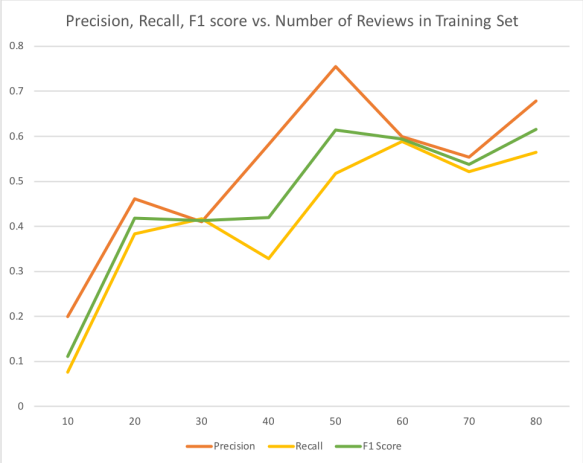

Figure 5: Accuracy measures vs Size of Training data

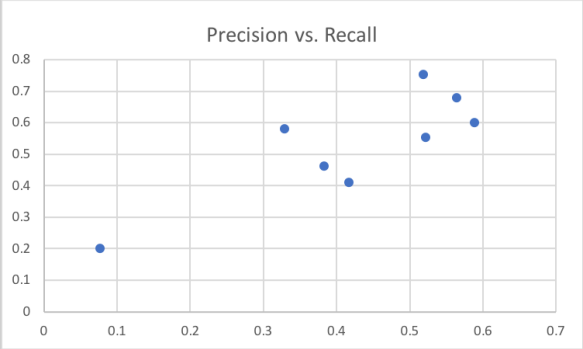

Figure 6: Precision vs Recall

The next step in our approach is to finalize a list of menu items using spaCy's similarity metrics and Python. Then, using the sentiment models created we can determine which menu items were the most popular for the restaurants in our test set. After this, we will implement a bidirectional LSTM with PyTorch, and perform similar calculations as in the spaCy model to generate the top lists.

7 Timeline for the rest of the project

- (1 week) Add additional annotations to build a stronger model.
- (Same week as annotating) Compile a menu through similarity metrics implemented with spaCy and Python. Potential similarity metrics include cosine similary, fuzzy matching, etc.
- (1 week) Implement the LSTM model with PyTorch and do the same as done for the NER model.

- (Half a week) Create a list of top menu items based on the sentiment models built and menu created by the model.
- (Half a week) Compare results and create poster.
- (One and a half weeks) Write up final draft.

References

References

[1] *Linguistic Features - SpaCy Usage Documentation*.
https://spacy.io/usage/linguistic-features

[2] *Evaluating spaCy Model*.
https://github.com/explosion/spaCy/blob/master/spacy/scorer.py

[3] *A Method for Automatic Extraction of Multiword Units Representing Business Aspects from User Reviews*.
https://doi.org/10.1002/asi.23052

[4] *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.
http://www.aclweb.org/anthology/N16-1030

[5] *Introducing Popular Dishes on Yelp: Taking the Guesswork Out of What to Order*.
https://www.yelpblog.com/2018/06/introducing-popular-dishes-on-yelp-taking-the-guesswork-out-of-what-to-order

[6] *Rousillon: Scraping Distributed Hierarchical Web Data*.
The 31st Annual ACM Symposium on User Interface Software and Technology

[7] *OxfordFoodDictionary*.
A Dictionary of Food and Nutrition

[8] *TensorFlow: A system for large-scale machine learning*.
https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf