Using Word Embedding to Automate Recipe Substitution

Jennifer Liu

Abstract

In this project, we explore the possibility of creating a model that can identify replacement for an ingredient. For instance, if the query was butter, it should discover that margarine is similar to butter, so that is 1 potential replacement. The applications for this are numerous: along with aiding individuals with allergies and dietary restrictions, this can also help cooks understand how to use ingredients they already have. The idea is to transform ingredients into embeddings. With ingredients represented as embeddings, cosine similarity can be used to find the closest replacement.

Introduction

Finding a substitute for experienced chefs is a trivial task. For an individual who has little to no experience with cooking, they often feel the need to follow the recipe exactly because they do not have enough knowledge about the potential substitutes for ingredients. An experienced chef would know that it is not always necessary to use the exact same ingredient. In fact, there are often substitutes that can make the food taste equally as good or even better. An experienced chef knows this through many trial and error in the kitchen. But, how can we leverage word embedding technique to automate this task? This paper explores this possibility.

Data

Data came from 2 sources: Kaggle Ingredient Dataset, and allrecipes common ingredient substitution list. There were 6,800 recipes from Kaggle Ingredient Dataset and 30 ingredients with replaceable suggestions from allrecipes.com common ingredient substitution page.

Data Preprocessing

From the collected JSON of all ingredients, we performed the following steps to clean the data.

1. For each item in the list, tokenize the words.
2. Lemmatize the ingredients. Hence, “tomatoes” normalizes to “tomato”
3. Created a list of measure words (such as “lb,” “cup”, “tsp”, etc), and check if the substitution contained a measure word. If it does, remove it.
4. After plurals are removed, spaces and hyphens were removed and replaced with underscores (turning “low-fat milk” into “low_fat_milk”)
5. Remove punctuation from ingredients if it exist.
6. Lowercase all ingredients.

A dictionary mapping an ingredient to a list of potential ingredients was created using the allrecipe common ingredient substitution page.

The input to generate a word vector was a list of list of ingredients. There were 2 different input vectors generated. Each of these vectors are a list of list of ingredients. While one considers the substitution lookup table and the raw ingredient , the other only parses the raw ingredient.

Parsing “Banana Coffee Cake with Pecans”

To better illustrate how the data is extracted, the following is a complete walk through of how a recipe is parsed. Given the recipe “Banana Coffee Cake with Pecans”, the parser returns a list of ingredients as shown below.

Ingredient List Result:
[superfine sugar, cup butter, eggs, baking soda, sour cream, mashed ripe banana, cake flour, baking powder, vanilla extract, pecans, brown sugar, ground cinnamon]. The vector will be referred to as word vectors without substitution.

Using the substitute lookup table generated, we find a potential substitution for each ingredient in the list. The list of substitutes for an ingredient like egg is merged with the ingredient list. Therefore, if the substitute for "egg" is ["baking_powder", "vinegar"], the final input for the word embedding for this recipe would be [all ingredients in the ingredient list result, "baking_powder", "vinegar"]

The reason that the ingredients are parsed as list of list rather than a single list is due to Gensim word embedding restrictions.

Challenges of Data Preprocessing
There were many problems when attempting to scrape recipes from AllRecipe.com substitute list.

First, The substitutions suggested by "allrecipe.com" are very descriptive. The recipe replacements can sometimes be a mixture of few different ingredients. For example, buttermilk can be replaced with "1 cup water plus 1/4 cup buttermilk powder". However, if we passed this as a substitution, the word embedding may have trouble finding similarities because it is so specific. Therefore, we decided to identify the main ingredients and use these as input. In this case, the main ingredients are water and buttermilk. Therefore, the resulting vector would be ['buttermilk', 'water', 'buttermilk_powder']. To identify these main ingredients, the sentence was parsed into their respective part of speech. Only nouns (NNS, NN) and adjectives (JJ) are considered as the main ingredient. One of the downsides of this method is that it considered "vegetable" and "oil" as two different ingredients. Initially, we decided that if two nouns occur consecutively in the phrase, it should be treated as a single word, but that gave us more false positives. Instead, we manually tweaked these ourselves.

Methods - Applying Vectors Semantics to Ingredient Substitutions
The idea was inspired by the fact that words that occurred together are likely to have the same meaning. This theory was applied to solve this problem. In this task, similar meaning is defined by the possible substitution. The hypothesis is that since the ingredients that are replaceable are placed in the same context, the embedding would show replaceable items as nearby each other. Similar to how word embeddings can learn that "apricot jam" is more likely to occur than "apricot aardvark" based on co-occurrence of "apricot", this system identifies that "milk" is more likely to occur with "evaporated milk" than it is with "apple" because they co-occur more frequently.

Based on this concept, four different types of vector representations were created:

1) Word2Vec using a list of ingredients that contained their potential substitution.
2) Word2Vec using only the list of ingredients scraped from Kaggle Dataset.
3) FastText using a list of ingredients that contained their potential substitution.
4) FastText using only the list of ingredients scraped from Kaggle Dataset.

Using the cosine similarity equation, we found the top 10 most similar ingredients given the requested ingredient for each of the word embedding generated using different vector representation. Then, the results were agglomerated to generate a single list. The agglomerated result was sorted based on the score, from highest to lowest.

The word embedding for the two FastText was given a lower weight. For each of the scores generated by the 2 FastText word vector, the weights were reduced by 20%. The weights for FastText was reduced because FastText considers word similarity from a character level. This caused problems for ingredients that did not have much substitutions. In these situation, it relied heavily on the spelling similarity rather than the culinary similarity. For instance, when finding substitute for "Margarine", it returned "Margarita Mix". After applying the weights, the system returned slightly more relevant replacements.

If it caused so much problem, why was FastText considered? The advantages of using FastText is that most of the time user may not input an ingredient that is spelled exactly the same way as our dataset. Without using FastText, the system may think that there contains no substitute for "eggs". But, the embedding does have substitute for "eggs", but it is in singular form. By using FastText, it is able to catch ingredients these types of minor mismatch in spelling between the user and our system.

Results

	Word2Vec with substitution	**FastText with substitution**	**Word2Vec without substitution**	**FastText without substitution**	**Best substitute – merged**
tofu	ricotta, dry cottage cheese, soybean paste, shimeji mushroom	Soft tofu, silken tofu, pressed tofu, tofu pureed	Soybean sprout, roasted sesame seed, soybean paste	Soft tofu, silken tofu, firm tofu, firm silken, tofu	Ricotta, soybean sprout, dry cottage cheese, enokitake, bean curd skin
butter	Soft wheat flour, icing, chopped pecan, single crust pie	Batter, vegan butter, cashew butter, bitter	Margarine, melted butter, butter oil, butter cooking spray	Batter, vegan butter, apple butter, butter oil	Batter, margarine, vegan butter, cashew butter
oregano	Ground oregano, cumin, smoked paprika, chorizo sausage	Oregano leaf, Mexican oregano, poblano, smoked paprika	Dried oregano, poblano pepper, parsley, pico de gallo	Oregano leaf, adobo seasoning, dried oregano leaves crushed	Ground oregano, cumin, dried oregano, oregano leaf

Table 1.0 – A table showing ingredients that produced culinary sound substitutes generated by the 4 different word embeddings. Of the top 10 best results generated from each embedding,the top 4 most reasonable ingredients were selected.

	Word2Vec with substitution	FastText with substitution	Word2Vec without substitution	FastText without substitution	Best substitute – merged
prawn	Spring onion, king prawn, chicken thigh fillet, peeled prawn	Tiger prawn, raw tiger prawn, spring onion, capsicum	Pak choi, king prawn, fresh red chilli, banana blossom	Raw king prawn, tiger prawn king prawn, lobster tail	Spring onion, minced pork, peeled prawn
rice	Brown rice, bulgur, barley, brown basmati rice	Minute rice, wild rice, black rice, Spanish rice	Long grain white rice, rice, long grain rice, instant rice	Instant white rice, minute rice, cooked white rice	Brown rice, bulgar, barley, wild rice

Table 2.0 – This table shows some of the ingredients that our system failed to produce culinary sound replacements.

Evaluation

It was extremely hard to evaluate a subjective topic such as this because there are so many different possible substitutes that are permissible. Mismatch between ingredients and labels does not necessitate an invalid replacement. Therefore, it is difficult to compute an accuracy and confusion matrix. With that said, a paper was found that showed a few of their results using a completely different method for finding substitution. The paper "Recipe Recommendation Using Ingredient Network", used 1,976,920 user's comment and graph theories to generate these substitution (Teng, 2012). We compared our substitution results with theirs and see how many matched up.

	Substitution result from (Teng, 2012)'s paper	Substitution using our method	Substitution from google searching
Olive oil	Butter, apple sauce, oil, banana, margarine	Vegetable oil, Pompeian canola oil, sun-dried tomato	Canola oil, peanut oil, sesame oil
Sweet potato	Yam, potato, pumpkin, butternut squash, parsnip	Yam, pumpkin, butternut squash	Yam, parsnip
Almond	Pecan, walnut, cashew, peanut, sunflower seed	Pistachio, hazelnut, apricot, prune, almond paste	Hazelnut, pistachio nuts, grape-nuts cereal

Table 3.0 – A table comparing the substitute generated by our system, and substitutes found online, and the substitute result from Teng et el's paper. (Teng et.al., 2012). Ingredients that were highlighted means that it either matched with a substitute found online or from the referenced paper. A maximum of 5 substitution ingredient were shown.

Every ingredient presented had at least one match with either the referenced paper. Mean reciprocal rank was considered during the evaluation process. However, it was unclear whether the online sources

used for the substitution lists were ranked in order. Moreover, at times, our system may not match with the referenced paper directly, but is still perfectly valid as a substitute. For instance, our system returned cauliflower as a substitute for spinach, and the reference paper did not. If we used mean reciprocal rank as an evaluation metric, it would have penalized our system even though it is valid.

Discussion

Compared to Teng et. al's work, our system did not have a particularly high performance. This is likely due to lack of good data. User comments was considered as the initial source of data. Nonetheless, parsing user comments from AllRecipe.com was more challenging than anticipated. User comments had a great deal of ambiguity. Thus, simply placing the rule "I substitute A with B" was not very effective.

Throughout the course of this project, an observation that was made was that substitution is a very subjective topic. Types of substitution you use is very dependent on what you make. For instance, replaceable item for "flour" can differ based on whether it is for making bread, cake, or self-rising. Our system currently agglomerates all the replacement into one category. In the case of flour, it considers all possible replacement for bread, cake, and self-rising.

Conclusion

There are ample amount of applications for a system like this. 1) An individual may not know what "tree nut" is, but it knows what "hazelnut" is. Because the system has given the user a comparison to an ingredient they never heard of before, they can get a better idea of what "tree nut" is. 2) This recipe replacement can help use up leftover food more effectively 3) It can help find substitutions for individuals with allergies.

An issue encountered during the data preprocessing stage was that there were some ingredients that contained brand names, while others did not. Consequently, the system would occasionally return two of the same kind of ingredient, but one would contain the brand name while the other did not. For instance, "Pompeian canola oil" and "Canola oil" are the same ingredient, but the word embedding does not think that due to the presence of "Pompeian". "Pompeian" is a brand for a canola oil.

We tried using the NLTK named entity recognition, but some of these brands were so specific to the cooking world, the NLTK package encountered difficulties recognizing these labels. For instance, "Pompeian" was a brand unrecognized by NLTK named entity recognition.

In the dataset, many of the ingredients were very specific, causing our system to generate "sticky rice" or "minute rice" as a substitute for "rice". These substitutions are not very helpful. But, if the more specific parts of the phrase were stripped, we would remove some actual ingredients. For instance, "rice vinegar" is not rice at all, but if the "vinegar" was stripped away the system would read it incorrectly. The biggest challenge encountered was that there does not exist a dataset that helped to establish culinary relationships.

In the future, we hope to improve the user interface by adding the capability to search for substitutions based on recipes, and allow users to input their specific restrictions (i.e peanut allergy). In terms of the method, we hope to collect more relevant data.

References

Teng, Chun-Yuen, et al. “Recipe Recommendation Using Ingredient Networks.” *Proceedings of the 3rd Annual ACM Web Science Conference on - WebSci '12*, 2012, doi:10.1145/2380718.2380757.

Huang, Steeve. “Word2Vec And FastText Word Embedding with Gensim.” *Towards Data Science*, Towards Data Science, 4 Feb. 2018, towardsdatascience.com/word-embedding-with-word2vec-and-fasttext-a209c1d3e12c.