Cooking Tools Retrival for Recipe Recommendation

Qinyu Feng, Bei Su, Kailin Zheng
Computer Science Department
New York University
New York, NY 10010, USA

Abstract

Recipe recommendation systems typically select the most appropriate recipe for users based on users' ingredients and cuisine preference. However, the cooking instructions sometimes cannot be followed when people do not have the required cooking tools. In this paper, we propose a cooking tools retrieval method that can extract necessary cooking tools according to the ingredients and instructions in the recipes. The source code of this project can be found on Github via this link: https://github.com/kz882/recipe

1 Introduction

Cooking recipes are collaboratively shared knowledge that can be easily searched up online. There are many commercialized APPs and websites available in the market that provides recipes recommendation for users based on ingredients search. For example, Supercook, a recipe search engine, categorizes ingredients into dairies, meat, vegetables, etc. and recommends recipes based on user input of ingredients. Normally a recipe would include ingredients and instructions. But they may not explicitly list the cooking tools necessary for the dishes.

However, cooking tools are as important, if not more, as ingredients. Sometimes necessary cooking tools such as aluminum foil and casserole dish may not be at hand upfront. In this situation, it would be better if the recipe recommendation systems could consider tools that the users have. To help with the recommendation, we develop a Cooking Tools Extraction System that extracts the required tools based on the nouns or noun groups that appear in the instructions. Our first approach is to use Part-Of-Speech tagging to extract the nouns and then eliminate the ingredients from the list of nouns. Our second approach is to compare the nouns with our knowledge base, a trained dictionary of cooking tools and deem them as results.

There are several challenges in the cooking tools retrieval task: first is that the current popular corpora used to train parsers contain news articles, which is under different domain comparing to cooking recipes. Therefore, the parsers do not work as well with our dataset. To resolve this issue, we implement domain adaptation. The second challenge is that it is difficult to decide whether words are related to food or not. For example, the "potato masher" is a named entity per se and the word "potato" here does not refer to an actual potato, but is an attribute to describe the masher. We did not implement any method to solve this issue due to time constraint, but we plan to use named entities in our future work. To evaluate our cooking tools retrieval method, we use precision, recall, and F-measure to compare the system output with the answers given by our human annotators.

2 Related Work

Mori et. al. convert the instructions of recipes as procedural texts into directed acyclic graphs as flow graphs to achieve a real "understanding" of the instructions despite the different expressions of human languages. They categorize the words in instructions into named entities (NEs) including food (F), tool (T), duration (D), quantity (Q), action by the chef (Ac), action by food (Af), state of foods (Sf) and state of tools (St). For our purpose, the NEs labeled "food" could be further analyzed and excerpted into ingredients, and the actions could be associated with cooking utensils.

Gunamgari et. al. present a more complex tagset for annotating recipes, which is hierarchical and recursive, with higher levels covering broader categories and lower levels covering specific tags. They propose more specific named entities, including Name of the Ingredient (NOI), Properties of Ingredients (POI), Total Size of the Ingredient (TSOI), Total Quantity of the Ingredient (TQOI), preprocessing actions performed on the Ingredients. Cooking steps are further divided into different specifications such as Cooking Action Specifications (ca), Utensils Specifications (uca), Utensils Specifications (uca), and Time Details (time).

Teng et. al. apply a more straightforward method to extract ingredients. They use regular expression matching to remove non-ingredient terms from the line and identify the remainder as the ingredient. They remove quantifiers, such as "1 lb" or "2 cups", words referring to consistency or temperature, such as "chopped" or "cold", along with a few other heuristics, such as removing the content in parentheses. For example, "1 (28 ounce) can baked beans (such as Bush's Original)" is identified as "baked beans". They then generate an ingredient list sorted by frequency of ingredient occurrence and select the top 1000 common ingredient names as their finalized ingredient list (Teng). We could use this method to more directly extract the essential information of ingredients to process our noun groups from instructions.

Agarwal and Duong apply Maximum entropy classification of named entities (MaxEnt classifier), semantic role labeling (SRL) and coreference resolution to "identify which actions are applied to which ingredients, and possibly identify which utensils are being used" (Agarwal). Since they only manually labeled 12 recipes for Named Entity Resolution (NER), "it becomes a major source of error for SRL, which in turn would cause errors in coreference resolution". As a result, having a more accurate NER would make the other pieces of their project more accurate. To generate more accurate NER results, our system seeks to improve the Part-Of-Speech tagging by domain adaptation from the Wall Street Journal to the recipes.

Honnibal uses Averaged Perceptron, which adapts the idea of supervised learning to apply a simple learning algorithm to assign different weights to features of each word and average them (Honnibal).

Daumé III proposes to leverage the large, annotated dataset from the source domain and a small, annotated dataset from the target domain. He adds more weight to the dataset of the target domain to achieve feature augmentation for domain adaptation (Daumé III).

3 Data

3.1 Recipe Box, Structured Recipes Scraped from Food Website

We download our dataset of ~125,000 recipes from Recipe Box, "Structured recipes scraped from food website" as JSON file from eightportions.com (Lee). The description of the dataset says that each recipe consists of: "a recipe title, a list of ingredients and measurements, instructions for preparation, and a picture of the resulting dish". Since the original dataset was scraped from various food websites, the JSON file con-

tains some unrelated content such as advertisements. We use regular expressions to remove trivial characters.

Example recipe file in JSON:

```
{'title': 'Slow Cooker Chicken and Dumplings',
  'ingredients': ['4 skinless, boneless chicken breast halves ADVERTISEMENT',
   '2 tablespoons butter ADVERTISEMENT',
   '2 (10.75 ounce) cans condensed cream of chicken soup ADVERTISEMENT',
   '1 onion, finely diced ADVERTISEMENT',
   '2 (10 ounce) packages refrigerated biscuit dough, torn into pieces ADVERTISEMENT',
   'ADVERTISEMENT'],
  'instructions': 'Place the chicken, butter, soup, and onion in a slow cooker, and fill with enough water to cover.\nCover, and cook for 5 to 6 hours on High. About 30 minutes before serving, place the torn biscuit dough in the slow cooker. Cook until the dough is no longer raw in the center.\n',
  'picture_link':
'55lznCYBbs2mT8BTx6BTkLhynGHzM.S'}
```

We read the JSON file into two Python dictionaries that contain titles as keys, ingredients and instructions as values, respectively. We divide our datasets into 50 recipes as the training set, 30 recipes as the development set and 20 recipes as the test set.

3.2 Training File (WSJ_02-21.pos) with Original POS Tags

The training file (WSJ_02-21.pos) consists of about 950K words. Each line consists of a token, a single blank, and the part-of-speech of that token using the Penn Treebank tag set. Sentence boundaries are marked by an empty line (about 40K sentences). The training corpora are retrieved from Wall Street Journal, so are mostly news articles about Finance. WSJ here is known as "source domain", which we will implement domain adaptation to improve the POS tagging accuracy for our recipe corpora, known as "target domain".

3.3 A Non-exhaustive List of Tools as the Knowledge Base

We compile a list of 170 tools retrieved from the following three websites: "List of food preparation utensils", "Kitchen Essentials List: 71 of the best kitchen cookware, utensils, tools & more" and "The Ultimate List of Kitchen Tools for Healthy Cooking". Each line contains a tool. For our purpose, we will only search the nouns in the knowledge base. We keep the adjectives and attributes in the list of tools for further work that considers more specific attributes of the tools.

4 Method and Implementation

4.1 Part-Of-Speech Tagging

Part-Of-Speech of a token refers to the role it plays in the syntactic function of a sentence. We use the "Alphabetical list of part-of-speech tags used in the Penn Treebank Project" as the definitions of our POS tags. We first apply the nltk.tag.perceptronpackage by Honnibal and Duong to the 50 recipes in the training set as the first step of POS tagging. Based on the POS tags given by the program, the annotators (two of our teammates) manually correct them. The nltk package tends to label the verbs as nouns. For example, in the instruction "Stir in the flour and salt, and simmer until bubbly," "Stir" is incorrectly labeled as a noun while "simmer" is labeled as a verb. This is because most of the sentences in WSJ are declarative whereas most of the instructions are imperative. If we use Viterbi algorithm (see Section 4.3), the first word in the sentence will be labeled as NN while in our case most of the first words in sentences are verbs. By correcting the noun/verb labels and other labels, we get more accurate POS tags and thus have a more accurate list of nouns for future processing.

4.2 Domain Adaptation by Feature Augmentation

We first use the Wall Street Journal corpora with POS tags as the training data. One issue with this dataset is that it contains sentences that may have different syntactic and mean-

ings and thus generate POS tags that are less reliable in the context of recipe instructions. In order to resolve this issue, we apply domain adaptation by adding manually corrected, POS tagged instructions of the 50 recipes to the Wall Street Journal corpora to generate a new training set. Since the size of the recipe dataset is not comparable to the WSJ corpora, we perform feature augmentation in the preprocessing step by adding the recipe data 100 times to make it equally weighted as the Wall Street Journal corpora, as suggested by Daumé III.

4.3 Viterbi Algorithm (Hidden Markov Model)

The Viterbi algorithm is a dynamic programming algorithm for finding the most likely sequence of hidden states—called the Viterbi path—that results in a sequence of observed events, especially in the context of Markov information sources and hidden Markov models (HMM) to generate POS tags based on training set ("Viterbi"). We label the development set by applying the Viterbi algorithm to the training set. We consider the POS tags of the previous word and following word and use conditional probability to calculate the likelihood of a particular word sequence.

$$P(W_n \mid W_{n-1}) = \frac{P(W_n, W_{n-1})}{P(W_{n-1})}$$

To apply the conditional probability to Viterbi, we follow the following steps:

1. Retrieve a list of observed words $W_1 \ldots W_t$ and a list of POS tags $Q_0 \ldots Q_f$ whereas Q_0 is the start state and Q_f is the end state.
2. Generate a NxN matrix A where A(i,j) is the probability that Q_j is the next POS tag to Q_i; generate a lookup table B where B(i,j) is the probability that POS tag j occurs to word i.
3. Calculate a Viterbi matrix where each element contains the score with maximum previous * prior * likelihood; meanwhile, note the backpointers as the assigned POS tags based on our algorithm.

5 Evaluation

From previous steps, our system generates a list of tools based on the noun groups in the instructions section of the recipes, known as "system output". The human annotators agree on a list of tools known as "answer key". For the two lists, we build a python program that automatically reads the inputs and compares the results based on the 20 recipes in the test set. For each recipe, we have a precision, recall, and f-1 score.

$$\text{Precision} = \frac{\text{number of correct tools}}{\text{number of system output}}$$

$$\text{Recall} = \frac{\text{number of correct tools}}{\text{number of answer keys}}$$

$$\text{F} - \text{measure} = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

6 Experiment

We take two different approaches toward selecting cooking tools from the noun list. The first method is to remove the ingredients and other non-cooking tool nouns from the list, and the second approach is to compare the list of nouns with our knowledge base tools. For the first method, we extract a list of ingredients and strip off the quantifiers, state of ingredients, etc from the ingredients list of our training set. For example, based off of the ingredients list of the sample JSON file, we expect to generate "chicken breast", "butter", "condensed cream", "onion", "biscuit dough". Then, we remove the ingredients from the list of nouns. The ingredient section of the recipe contains the list of food and quantity needed for the dish. Since food is included in the noun list that we generated, we remove them from the list to eliminate the incorrect answers in the list. And for the second method, we compare the list of nouns with our knowledge base, a trained dictionary of cooking tools and deem as results.

Sample input for recipe #51:

```
51 Creamy Au Gratin Potatoes
Preheat oven to 400 degrees F (200 de-
grees C). Butter a 1 quart casserole
dish.
Layer 1/2 of the potatoes into bottom of
the prepared casserole dish. Top with
the onion slices, and add the remaining
```

```
potatoes. Season with salt and pepper to
taste.
In a medium-size saucepan, melt butter
over medium heat. Mix in the flour and
salt, and stir constantly with a whisk
for one minute. Stir in milk. Cook until
mixture has thickened. Stir in cheese
all at once, and continue stirring until
melted, about 30 to 60 seconds. Pour
cheese over the potatoes, and cover the
dish with aluminum foil.
Bake 1 1/2 hours in the preheated oven.
```

Sample output of approach I:

sys_output.txt

```
51      Creamy Au Gratin Potatoes
oven
saucepan
seconds

52      Garlic Chicken
oven
saucepan
bowl
instant-read
```

Sample output of approach II:

sys_output_2.txt

```
51      Creamy Au Gratin Potatoes
oven
dish
saucepan
foil

52      Garlic Chicken
oven
saucepan
bowl
dish
oven
```

Answer key by annotators:

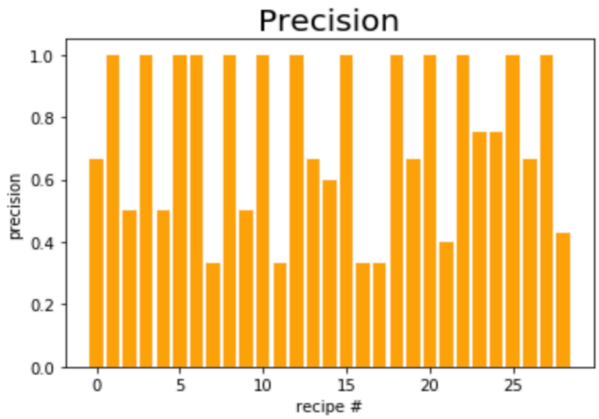
answer_keys.txt

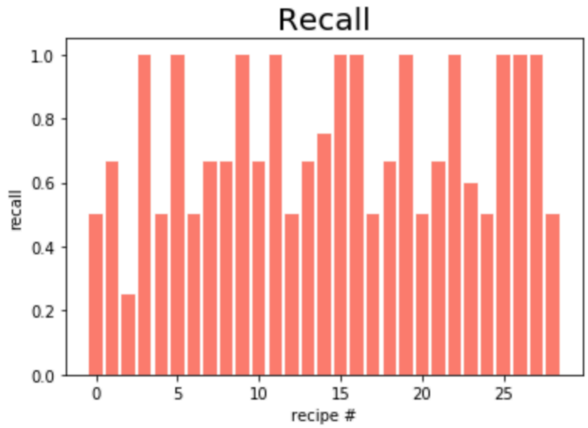
```
51      Creamy Au Gratin Potatoes
oven
saucepan
whisk
aluminum foil

52      Garlic Chicken
oven
saucepan
shallow bowl
tong
baking dish
instant-read thermometer
```

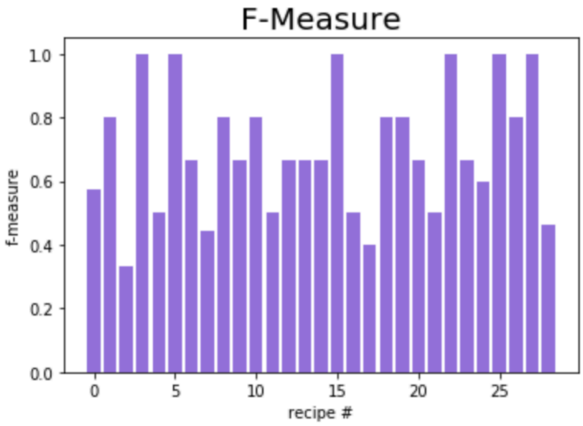
7 Results and Discussion

Below we show the precision, recall, and F-score for the cooking tool list based on the two different approaches. These data were computed for a fixed training set of 50 recipes, development set of 30 recipes, and 30 testing recipes. Each recipe contained about 5-10 ingredients, approximately 10 sentences of instructions, and 3-5 different cooking tools

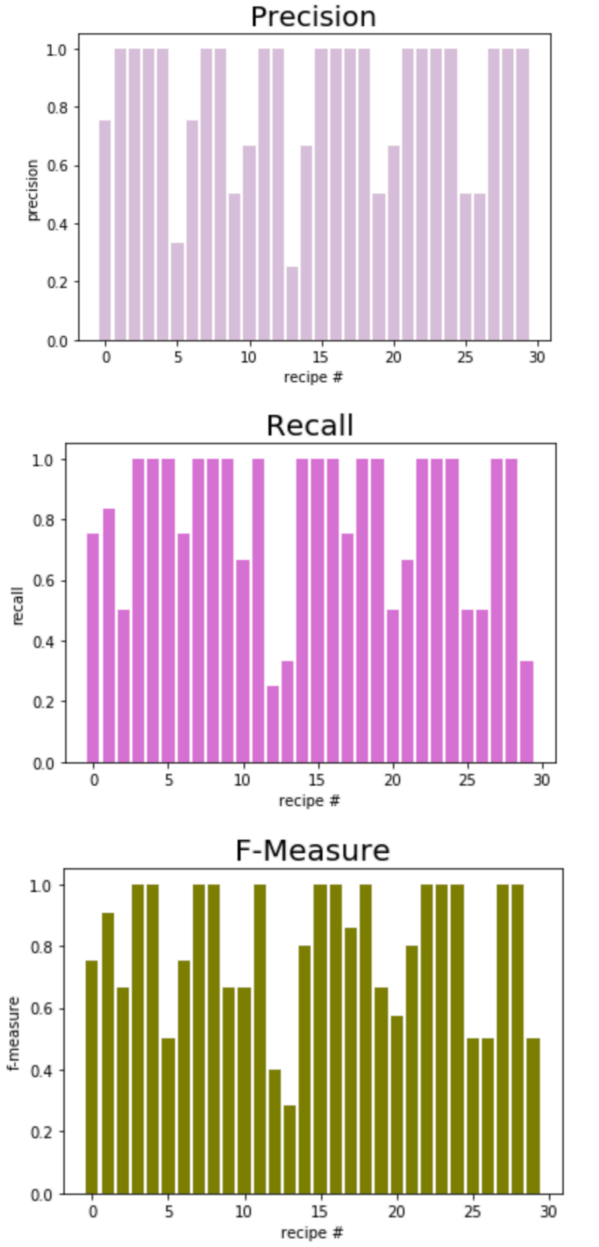
Approach I:







Approach II:



Approach	Precision	Recall	F-measure
1	0.725	0.720	0.687
2	0.836	0.811	0.793

The evaluation result shows that our second approach yields a higher precision, recall, and F-measure than the first approach. This maybe because human annotators tend to come up with a list of tools they know instead of eliminating ingredients from the noun groups. Therefore, the second approach is more natural and closer to human habit. However, the result is somewhat misleading, since we neglect the adjectives and only consider nouns, but our answer keys contain the complete noun group, including both the adjective and the noun, such as slower cooker, backing dish etc. So we consider our system output to be correct even if it only contains the trimmed version, and matches as the substring of the corresponding answer key. In fact, the trimmed version may not be correct. In that way, the tools in both our answer key and system output would be simplified, leading to a higher score.

8 Future Work

Due to time constraint, several other approaches that can potentially optimize our algorithm are not implemented in the current version of our system.

We would like to generate a more exhaustive list of ingredients and tools for a better knowledge base. Our current system strips the ingredients completely off of the adjectives. However, to prepare the ingredients, some tools may be necessary to pre-process the ingredients before we move on to the instructions section. For example, in the sample JSON recipe file, we may need boning knife and peeling knife to strip chicken breast off of bones and skin. We may also need a special knife to "finely dice" the onion, and use the fridge to freeze the biscuit dough, etc. So the section of ingredients should also be considered in addition to the instructions.

We can use BIO tagging to specify the nouns to consider adjectives and quantifiers. BIO tag refers to Inside-outside-beginning tagging that groups the noun groups together. For example, the slower cook is more informative than cooker; a 12 inches pan is more specific than simply "pan", since "pan" can refer to frying pan or other types of pan; the sheet could be baking sheet, cookie sheet, etc. We lose information if we omit the attributes whereas BIO tagging provides us with more complete information.

Now we have retrieved ingredients based on the nouns in the instructions; an updated system will speculate the tools based on the action verbs. We will first pre-train the list of tools (our current knowledge base) to associate each tool with a list of potential action verbs. To do this, we can scrape the description of the tools from their respective Wikipedia page or definitions in the Oxford English. We will calculate the percentage of words that occur together in the same file, or that occur in the same sentence. Then we will associate the tool with the top few action verbs in its description and obtain the Pairwise Mutual Information. Alternatively, we can use the Google BERT system to do this task with machine learning via word-embedding.

References

Alphabetical list of part-of-speech tags used in the Penn Treebank Project:

https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

Agarwal, Rahul. Miller, Kevin. “Information Extraction from Recipes.”

https://nlp.stanford.edu/courses/cs224n/2011/reports/rahul1-kjmiller.pdf

Chun-Yuen Teng, Yu-Ru Lin, and Lada A. Adamic. 2012. Recipe Recommendation Using Ingredient Networks. In Proceedings of the 3rd Annual ACM Web Science Conference, ACM, pages. 298-307.

Daumé III, Hal. “Frustratingly Easy Domain Adaptation”. 2009. https://arxiv.org/pdf/0907.1815.pdf

Gunamgari, Sharath Reddy, Sandipan Dandapat, and Monojit Choudhury. "Hierarchical Recursive Tagset for Annotating Cooking Recipes." In Proceedings of the 11th International Conference on Natural Language Processing, pp. 353-361. 2014.

Honnibal, Matthew. Duong, Long. Source code for nltk.tag.perceptron.

https://www.nltk.org/_modules/nltk/tag/perceptron.html

Honnibal, Matthew. A Good Part-of-Speech Tagger in about 200 Lines of Python. https://explosion.ai/blog/part-of-speech-pos-tagger-in-python

Kitchen Essentials List: 71 of the best kitchen cookware, utensils, tools & more. https://www.mealime.com/kitchen-essentials-list

Lee, Ryan T. Personal Data Science blog “Eight Portions”project. https://eightportions.com/datasets/Recipes/#fn:1

N. Shino, R. Yamanishi and J. Fukumoto, "Recommendation System for Alternative-Ingredients Based on Co-occurrence Relation on Recipe Database and the Ingredient Category," 2016 5th IIAI International Congress on Advanced Applied Informatics (IIAI-AAI), Kumamoto, 2016, pp. 173-178.

Shinsuke Mori, Hirokuni Maeta, Yoko Yamakata, Tetsuro Sasada. “Flow Graph Corpus from Recipe Texts”. Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14). 2014. European Language Resources Association (ELRA) Pp. 2370–2377.

SuperCook. https://www.supercook.com/#/recipes

Ueda, Mayumi & Asanuma, Syungo & Miyawaki, Yusuke & Nakajima, Shinsuke. (2014). Recipe Recommendation Method by Considering the User's Preference and Ingredient Quantity of Target Recipe. Lecture Notes in Engineering and Computer Science. 2209. 519-523.

The Ultimate List of Kitchen Tools for Healthy Cooking.https://greatist.com/eat/ultimate-list-kitchen-tools-healthy-cooking#1

“Viterbi Algorithm.” *Wikipedia*, Wikimedia Foundation, en.wikipedia.org/wiki/Viterbi_algorithm.

Wikipedia page. List of food preparation utensils.https://en.wikipedia.org/wiki/List_of_food_preparation_utensils