Term Set Expansion In Food Recipe Using Context Measurements Including Bag-Of-Words and Symmetric Patterns

Anona Yang yy2956@nyu.edu
Jakub Podmokly jakub.podmokly@nyu.edu
Jennie Wang rw2364@nyu.edu

Natural Language Processing Class
Fall 2021
Professor Adam Meyers

Abstract

We present a combined approach for expanding terminology sets, which focuses specifically on words sharing a significant aspect of their meaning in food recipes when providing particular word seeds. We preprocess and tokenize the input corpus to determine any potential candidates for terms. On such prepared input we use context measurements, specifically a combination of bag-of-words and symmetric pattern methods to evaluate the semantic similarity between potential terms. We integrate the separate measurements by combining the resulting cosine similarity scores between term candidates and predefined elements of the seed set. We evaluate our algorithm by data from knowledge graphs, using Mean Average Precision (MAP). Our fully unsupervised results are satisfying and meaningful, compared to previous works that used a less specific data corpus.

1 Credits

This paper is written by Anona Yang, Jakub Podmokly and Jennie Wang (alphabetically ordered), receiving instruction from Professor Adam Meyers and mentor Shubham Vatsal.

2 Introduction

We often encounter situations in which we need to quickly gather the related key words from a particular category. If done manually, tasks can quickly become very time-consuming and may require deep familiarity with the context of particular terminology. This is when Term Set Expansion (TSE) (Kushilevitz et al., 2020) comes into sight. Given a small seed set of example terms from a semantic class, TSE is capable of finding more members of that class from a given data set. It enables people to easily select a seed set of terms and continuously expand it, thus simplifying the extraction of domain-specific fine-grained semantic classes. It can be used in different scenario of searching information, saving time and improving efficiency. For our project, we construct a Term Set Expansion system for food categories based on data obtained from cooking recipes. When planning a meal, one usually does well when strictly following a recipe but may struggle if some of the listed ingredients are unavailable. In such situations, knowing a popular alternatives to some specific items may be very helpful; similarly, when altering the recipes due to allergies or dietary preferences, tools like this one can become very handy.

Our goal in this paper is a fully unsupervised discovery of food categories from menu data sets. Typical methods include bag-of-words, symmetric patterns, and dependency relations. Our approach is based on a combination of bag-of-words and symmetric patterns, and we will utilize the following stages to complete our system:

1. The training data is preprocessed by isolating instruction sentence sets from the recipes, which contents are later tokenized and appropriately tagged.

2. Tokenized sentences are then used to train a word2vec algorithm that allows for a bag-of-words analysis, as well as a symmetric pattern search.

3. Such defined measurements provide cosine similarity scores between seed terms and candidate terms, which can be merged to establish a properly expanded set.

The performance of our system has been evaluated by calculating the Mean Average Precision. The MAP scores have been compared with the results in previous works.

3 Previous Work

According to Kushilevitz et al., solving a TSE problem requires a two-part algorithm (Kushilevitz et al., 2020). The first one identifies the desired concept class based on few examples, while the second one identifies additional members of the class. Previous attempts of solving this issue can be categorized into distributional and pattern-based approaches. The distributional approach operates based on the hypothesis that similar words appear in similar contexts. It is implemented by defining each term from the vocabulary as an embedding vector that summarizes all different contexts in which the term appears in in the corpus. Then, it looks for terms with vectors that are similar to those of the seed terms. On the contrary, the pattern-based approach considers specific indicative patterns that signal the desired concept class of words, looking for them in a large corpus, and extracting the terms that appear in them. Kushilevitz et al. combined these two approaches, considering only specific, indicative corpus locations and using the distributional nature of the neural LM to generalize across patterns and corpus locations (Kushilevitz et al., 2020).

Mamou et al. approached the TSE problem by implementing several distributional methods of context measurement. They constructed term context-value vectors according to different value categories in which target class of words may appear (Mamou et al., 2018). The context measurements they mentioned include Linear Bag-of-Words Context, Explicit Lists, Syntactic Dependency Context (Dep), Symmetric Patterns (SP), and Unary Patterns (UP). They then trained a Multilayer Perceptron classifier to predict whether a candidate term should be part of the expanded set based on ten similarity scores obtained by the above mentioned five different context types and similarity-scoring methods. The similarity scores are calculated and estimated by the cosine similarity between the centroid of the seed terms and each candidate term, and by the average pairwise cosine similarity between each seed term and each candidate term. Their approach is reachable from our perspective and would then be a main reference for our approach, which will be discussed in the next section.

Roark and Charniak (Roark and Charniak, 1998) discuss how to improve accuracy and efficiency when searching for a word category by incorporating the co-occurrence statistics of nouns. Selecting the initial seed word from among the most frequent head nouns in the corpus would increase the system performance, which is a great idea we will use for choosing our seed words.

In *Distributed Representations of Words and Phrases and their Compositionality* (Mikolov et al., 2013), authors discuss different ways of improving the quality of the vectors and the training speed by focusing on continuous Skip-gram model. In order to deal with the least frequent words effectively, this article claims that sub-sampling of frequent words during training results in a significant speedup (around 2x - 10x), and improves accuracy of the representations of less frequent words. We can use this method to avoid errors associated with rare and sparse categories, while also speeding up the overall system.

In *Terminology Extraction with Term Variant Detection*(Cram and Daille, 2016), authors talk about TermSuite, which follows the classic two steps of terminology extraction tools and the identification of term candidates and their ranking. This article explains the context and programming works for the term set expansion and terminology extraction. We have been inspired to incorporate several of the described programming methods in our solution, as well as compare the mathematical methods they have used. As we would like to compare the effectiveness of the dependency relations measurements between potential terms and other words present in the data set, these probability and theoretical calculations are an important part of a complete analysis.

4 Algorithm Overview

To successfully expand the term sets that would match the provided seed terms, our solution needs to incorporate several stages of data processing. (Figure 1)

Firstly, the algorithm needs to determine which elements constituting the input data can be considered to be potential terms. Due to the characteristics of our problem and the associated data, we have decided that noun phrases will be a good potential estimation for localizing term values. As our program focuses on analyzing cooking recipes, term categories (e.g., vegetables, spices, liquids) should be contained within the ranges defined by noun phrases (Mamou et al., 2018). To determine the presence of noun phrases in text, we use nltk

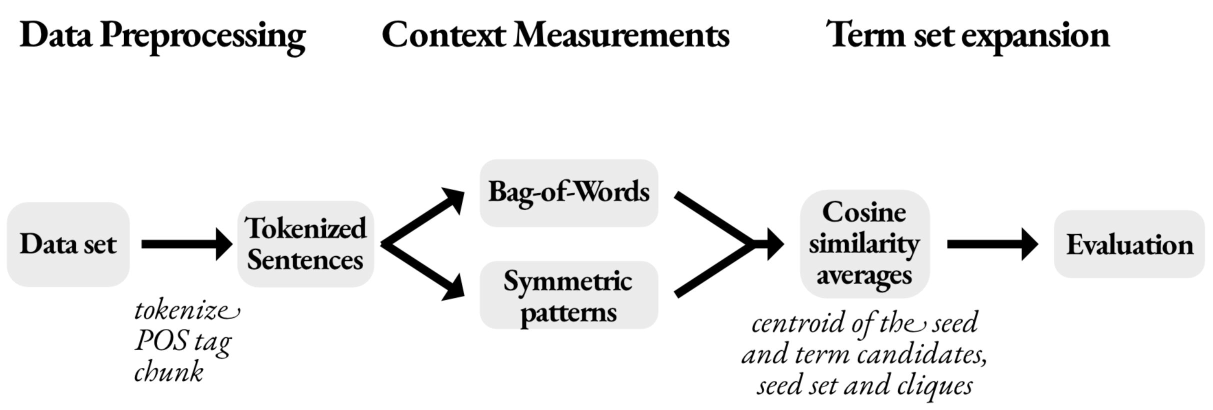


Figure 1: Pipeline of Our Approach

provided tokenizer, POS tagger, and chunker.(nlt)

Basing on the assumption that semantically similar words would appear in similar contexts, we estimate the semantic similarity using two different lexical context measurements (Mamou et al., 2018). As the first context measurement, we use the method of bag-of-words neighboring context units association, by incorporating the word2vec method (Mikolov et al., 2013). For this task, we applied the implementation of the word2vec vector training model provided by the gensim library (gen) on our preprocessed data.

Additionally, semantically similar terms can often appear in text in symmetric patterns (Mamou et al., 2018). Syntactic formations such as comma separated lists (Roark and Charniak, 1998), conjunction and disjunction (e.g., 'A and B', 'A or B'), or comparisons (e.g., 'A than B') (Davidov and Rappoport, 2006), are great examples of symmetric patterns, which communicate strong semantic similarity on a particular axis between the involved terms (Mamou et al., 2018). In the context of our problem, similar terms may often be listed together (e.g., salt and pepper, onion and garlic), allowing us to spot and measure the similarity between them. To not limit our approach to strictly predefined instances of symmetric patterns, we implemented the system used by Davidov and Rappoport, 2006, which enables us to learn patterns that demonstrate symmetric characteristics directly from our data. The result of such implementation are strongly connected 2-cliques, establishing broad and specific categories which portray relation between words.

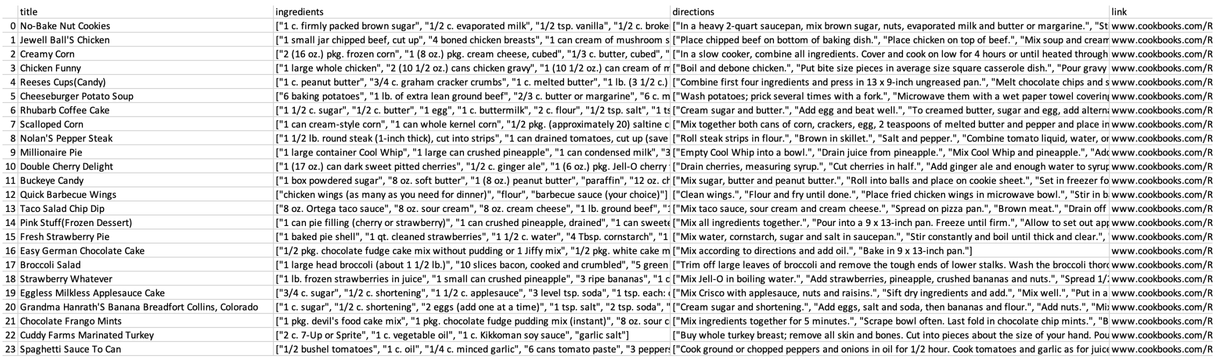
To prepare different context measurement values for further processing, term similarity is evaluated in two separate ways. For bag-of-words measurement, potential terms have associated cosine similarity between their modeled vectors and the vector of the centroid of the seed set (Mamou et al., 2018). For symmetric patterns measurement, the cosine similarity is calculated from vectors incorporating the amount of seed terms included in the clique relative to the size of the clique, between the term set and each clique. We summarize the similarities from such defined measurements through an average, prioritizing the bag-of-words measurement and incorporating smoothing to account for words not included in both measurements.

5 Data Preparation

The data we use to train our system incorporate over 2 million cooking recipes from RecipeNLG (Rec). These recipe data are divided into two sections, the first one including a list of ingredients, followed by a section of instructions (Figure 2). The instructions are provided as independent sentences with no co-reference between them, which allows us to only focus our analysis on word-level and sentence-level features, instead of analyzing semantic values that expand over individual sentences. Figure 2 is a general view of our data set and Figure 3 is an example of one instance of a recipe.

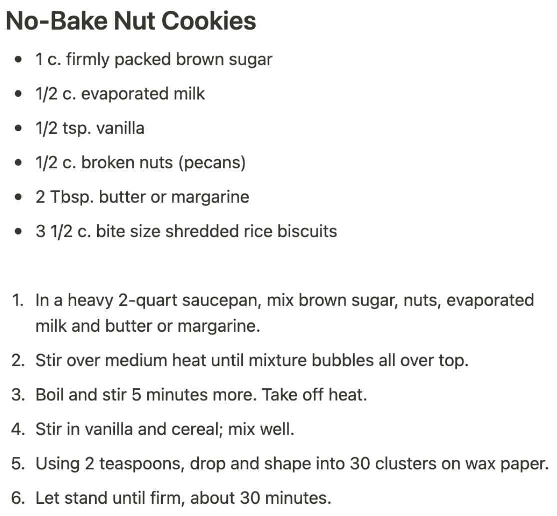
Such formatting of the data allows us to better incorporate general measurement methods such as short span bag-of-words and symmetric patterns, which yield great potential to characterize different food words.

For the preparation of the data, we POS tagged and tokenized them by the *nltk* package in Python.



	title	ingredients	directions	link
0	No-Bake Nut Cookies	["1 c. firmly packed brown sugar", "1/2 c. evaporated milk", "1/2 tsp. vanilla", "1/2 c. broke	["In a heavy 2-quart saucepan, mix brown sugar, nuts, evaporated milk and butter or margarine.", "St	www.cookbooks.com/R
1	Jewell Ball'S Chicken	["1 small jar chipped beef, cut up", "4 boned chicken breasts", "1 can cream of mushroom s	["Place chipped beef on bottom of baking dish.", "Place chicken on top of beef.", "Mix soup and crean	www.cookbooks.com/R
2	Creamy Corn	["2 (16 oz.) pkg. frozen corn", "1 (8 oz.) pkg. cream cheese, cubed", "1/3 c. butter, cubed", "	["In a slow cooker, combine all ingredients. Cover and cook on low for 4 hours or until heated through	www.cookbooks.com/R
3	Chicken Funny	["1 large whole chicken", "2 (10 1/2 oz.) cans chicken gravy", "1 (10 1/2 oz.) can cream of m	["Boil and debone chicken.", "Put bite size pieces in average size square casserole dish.", "Pour gravy	www.cookbooks.com/R
4	Reeses Cups(Candy)	["1 c. peanut butter", "3/4 c. graham cracker crumbs", "1 c. melted butter", "1 lb. (3 1/2 c.)	["Combine first four ingredients and press in 13 x 9-inch ungreased pan.", "Melt chocolate chips and s	www.cookbooks.com/R
5	Cheeseburger Potato Soup	["6 baking potatoes", "1 lb. of extra lean ground beef", "2/3 c. butter or margarine", "6 c. m	["Wash potatoes; prick several times with a fork.", "Microwave them with a wet paper towel covering	www.cookbooks.com/R
6	Rhubarb Coffee Cake	["1 1/2 c. sugar", "1/2 c. butter", "1 egg", "1 c. buttermilk", "2 c. flour", "1/2 tsp. salt", "1 ts	["Cream sugar and butter.", "Add egg and beat well.", "To creamed butter, sugar and egg, add alterna	www.cookbooks.com/R
7	Scalloped Corn	["1 can cream-style corn", "1 can whole kernel corn", "1/2 pkg. (approximately 20) saltine c	["Mix together both cans of corn, crackers, egg, 2 teaspoons of melted butter and pepper and place in	www.cookbooks.com/R
8	Nolan'S Pepper Steak	["1 1/2 lb. round steak (1-inch thick), cut into strips", "1 can drained tomatoes, cut up (save	["Roll steak strips in flour.", "Brown in skillet.", "Salt and pepper.", "Combine tomato liquid, water, or	www.cookbooks.com/R
9	Millionaire Pie	["1 large container Cool Whip", "1 large can crushed pineapple", "1 can condensed milk", "3	["Empty Cool Whip into a bowl.", "Drain juice from pineapple.", "Mix Cool Whip and pineapple.", "Add	www.cookbooks.com/R
10	Double Cherry Delight	["1 (17 oz.) can dark sweet pitted cherries", "1/2 c. ginger ale", "1 (6 oz.) pkg. Jell-O cherry	["Drain cherries, measuring syrup.", "Cut cherries in half.", "Add ginger ale and enough water to syrup	www.cookbooks.com/R
11	Buckeye Candy	["1 box powdered sugar", "8 oz. soft butter", "1 (8 oz.) peanut butter", "paraffin", "12 oz. ch	["Mix sugar, butter and peanut butter.", "Roll into balls and place on cookie sheet.", "Set in freezer fo	www.cookbooks.com/R
12	Quick Barbecue Wings	["chicken wings (as many as you need for dinner)", "flour", "barbecue sauce (your choice)"]	["Clean wings.", "Flour and fry until done.", "Place fried chicken wings in microwave bowl.", "Stir in ba	www.cookbooks.com/R
13	Taco Salad Chip Dip	["8 oz. Ortega taco sauce", "8 oz. sour cream", "8 oz. cream cheese", "1 lb. ground beef", "1	["Mix taco sauce, sour cream and cream cheese.", "Spread on pizza pan.", "Brown meat.", "Drain off	www.cookbooks.com/R
14	Pink Stuff(Frozen Dessert)	["1 can pie filling (cherry or strawberry)", "1 can crushed pineapple, drained", "1 can sweete	["Mix all ingredients together.", "Pour into a 9 x 13-inch pan. Freeze until firm.", "Allow to set out app	www.cookbooks.com/R
15	Fresh Strawberry Pie	["1 baked pie shell", "1 qt. cleaned strawberries", "1 1/2 c. water", "4 Tbsp. cornstarch", "1	["Mix water, cornstarch, sugar and salt in saucepan.", "Stir constantly and boil until thick and clear.", "	www.cookbooks.com/R
16	Easy German Chocolate Cake	["1/2 pkg. chocolate fudge cake mix without pudding or 1 Jiffy mix", "1/2 pkg. white cake m	["Mix according to directions and add oil.", "Bake in 9 x 13-inch pan."]	www.cookbooks.com/R
17	Broccoli Salad	["1 large head broccoli (about 1 1/2 lb.)", "10 slices bacon, cooked and crumbled", "5 green	["Trim off large leaves of broccoli and remove the tough ends of lower stalks. Wash the broccoli thoro	www.cookbooks.com/R
18	Strawberry Whatever	["1 lb. frozen strawberries in juice", "1 small can crushed pineapple", "3 ripe bananas", "1 c	["Mix Jell-O in boiling water.", "Add strawberries, pineapple, crushed bananas and nuts.", "Spread 1/2	www.cookbooks.com/R
19	Eggless Milkless Applesauce Cake	["3/4 c. sugar", "1/2 c. shortening", "1 1/2 c. applesauce", "3 level tsp. soda", "1 tsp. each: (	["Mix Crisco with applesauce, nuts and raisins.", "Sift dry ingredients and add.", "Mix well.", "Put in a	www.cookbooks.com/R
20	Grandma Hanrath'S Banana Breadfort Collins, Colorado	["1 c. sugar", "1/2 c. shortening", "2 eggs (add one at a time)", "1 tsp. salt", "2 tsp. soda", "	["Cream sugar and shortening.", "Add eggs, salt and soda, then bananas and flour.", "Add nuts.", "Mix	www.cookbooks.com/R
21	Chocolate Frango Mints	["1 pkg. devil's food cake mix", "1 pkg. chocolate fudge pudding mix (instant)", "8 oz. sour c	["Mix ingredients together for 5 minutes.", "Scrape bowl often. Last fold in chocolate chip mints.", "B	www.cookbooks.com/R
22	Cuddy Farms Marinated Turkey	["2 c. 7-Up or Sprite", "1 c. vegetable oil", "1 c. Kikkoman soy sauce", "garlic salt"]	["Buy whole turkey breast; remove all skin and bones. Cut into pieces about the size of your hand. Pou	www.cookbooks.com/R
23	Spaghetti Sauce To Can	["1/2 bushel tomatoes", "1 c. oil", "1/4 c. minced garlic", "6 cans tomato paste", "3 peppers	["Cook ground or chopped peppers and onions in oil for 1/2 hour. Cook tomatoes and garlic as for juice	www.cookbooks.com/R

Figure 2: Dataset



No-Bake Nut Cookies

- 1 c. firmly packed brown sugar
- 1/2 c. evaporated milk
- 1/2 tsp. vanilla
- 1/2 c. broken nuts (pecans)
- 2 Tbsp. butter or margarine
- 3 1/2 c. bite size shredded rice biscuits

1. In a heavy 2-quart saucepan, mix brown sugar, nuts, evaporated milk and butter or margarine.
2. Stir over medium heat until mixture bubbles all over top.
3. Boil and stir 5 minutes more. Take off heat.
4. Stir in vanilla and cereal; mix well.
5. Using 2 teaspoons, drop and shape into 30 clusters on wax paper.
6. Let stand until firm, about 30 minutes.

Figure 3: Recipe Example

6 Our System

6.1 Context Measurements

We used two context measurement methods to find the best potential terms to expand our seed set. While our Bag-of-Words approach is a customized application of word2vec vector training model provided by the gensim library (gen), the symmetric patterns measurement involves our own implementation of the method designed by Davidov and Rappoport, 2006.

6.1.1 Bag-of-Words

The Word2Vec model (gen) involves a recent approach that uses a shallow neural network to embed words in a particularly defined vector space. Such created word vectors allow for comparisons between words, holding similar values for words of similar meaning based on context. This specific implementation is based on a skip-gram model, which analyzes words by through a 2-word window and training 1-hidden-layer neural network, which attempts to successfully predict the probability distribution of nearby words (gen). In our implementation, the resulting vectors have size 150, which turned out to be a perfect middle-ground between precision and optimization.

6.1.2 Symmetric Patterns

To assess the measurement for symmetric patterns, we followed the methods described in *Efficient Unsupervised Discovery of Word Categories Using Symmetric Patterns and High Frequency Words*, (Davidov and Rappoport, 2006). To be specific, our algorithm follows the steps below:

1. In a first loop of the whole corpus, count the occurrences of each word. Set up T_H and T_C, identify high frequency word (HFW or H) and content word (CW or C) according to the average occurrences, where a HFW is a word appearing more than T_H times per million words, and a CW is a word appearing less than T_C times per a million words. We

```
For fruit ['orange', 'peach']:
['pear', 'delicata', 'nectarine', 'fig', 'buttercup', 'plum', 'quince']
For spice ['allspice', 'basil']:
['oregano', 'thyme', 'ginger', sage', 'cayenne', 'rosemary', 'coriander']
For meats ['deer', 'sausage']:
['poultry', 'filet', 'elk', 'shoulder', 'slice', 'carve', 'veal']
```

Figure 4: Sample term outputs for particular seed queries provided by the bag-of-words word2vec model. Output words are ordered by cosine similarity scores.

```
Short clique:
['grapefruit', 'kiwi', 'vitamin', 'currant', 'nutrient', 'calcium']

Longer clique:
['tequila', 'nacho', 'lemonade', 'citrus', 'croissant', 'specialty',
'poultry', guacamole', 'beverage', 'juicer', 'nutella', 'breakfast',
'kitchenaid', morning', 'marmalade', 'prawn','risotto', 'crabmeat', 'lunch',
'aubergine', 'chardonnay', 'tostada', 'nutribullet', ...]
```

Figure 5: Sample cliques resulting from our symmetric pattern analysis. Some of them were short, while multiple sets were considerably longer.

used $T_H = 100, T_C = 5$

2. Define meta-pattern: $HCHC$, $CHCH$, CHC, $CHHC$. Define: word X and Y are nodes, $X \rightarrow Y$ is an arc in the graph dictionary if and only if X and Y exist in the same pattern and X precedes Y. In a second loop of the whole corpus, identify existing meta-patterns, record the occurrences of all the patterns, remove patterns that appear in the corpus less than T_P times per million words; store the arcs according to the meta-patterns inside the graph dictionary. We used $T_P = 20$.

3. Calculate M_1, M_2, and M_3 for each pattern, , where

   (a) M_1 = the proportion of words that can appear in both slots of the pattern,

   (b) M_2 = the proportion of the number of symmetric nodes in the graph dictionary,

   (c) M_3 = the proportion of the number of symmetric nodes in the graph dictionary.

   They yield values in $[0, 1]$, a higher value indicates more symmetry. List the results as a sorted list from highest to lowest for each parameter.

4. According to results we calculated in the previous step, remove patterns that are not in the top Z_T in any of the three lists, and patterns that are in the bottom Z_B in at least one of the lists. We used $Z_T = 100$, $Z_B = 100$.

5. In a third loop of the whole corpus, for each kept pattern find all strong 2-cliques (that is for two words X and Y, we have $X \leftrightarrow Y$). Store corresponding categories. (For example, if A, B, C, D, E are nodes and $A \leftrightarrow B$, $A \leftrightarrow C$, $A \leftrightarrow D$, $A \leftrightarrow E$, $B \leftrightarrow C$, $B \leftrightarrow D$, $C \leftrightarrow E$, then we have two separate categories $\{A, B, C, D\}$, $\{A, C, B, E\}$);

6. Merge categories we had from the last step if and only if there's more than a 50% overlap between two categories, make sure they contain nouns as defined by the initial POS tagging.

6.2 Getting the Results

To incorporate the results from both context measurement techniques, we calculate the average of cosine similarity scores for the Bag-of-Words and symmetric patterns results. For the Bag-of-Words, cosine similarity is calculated between the centroid vector of the seed set and each of the potential terms, with the highest values selected for further analysis (Figure 4). For the symmetric patterns, the cliques that include full or partial members of the seed set are analyzed in terms of additional potential members of the category, taking into account the length of the seed set, the length of the clique, and the overlapping of the terms between the seed set and the clique. Individual members of the clique are then assigned specific similarity scores based on the cosine similarity of the sets (Figure 5). Finally, we incorporate smoothing to the similarity values to account for words that might not be included in both measurements. Examples of different outputs for particular seed sets are included in Figure 6.

7 Evaluation

7.1 Data and Corpora

We obtained predefined term sets describing food categories from knowledge graphs (https://conceptnet.io/), including separated sets of spices, fruit, and meat category members of an average length of 37 (Figure 7). All of the terms are associated with their respective category through a 'IsA' relationship within the knowledge graph. As the output of our system is a ranked list of possible words according to their possibilities to fit in the embedding vector, it is important for us to focus on the general precision. We have also made sure that the terms from the knowledge graph can be found in our data set, which resulted in an average cut of 4% of the original evaluation term sets of words not occuring in the recipe data.

7.2 Thresholds, Statistics and Examples

The performance of our system was evaluated by the Mean Average Precision at different top n values (MAP@n): MAP@10, MAP@20 and MAP@50. From the evaluation term sets, we have randomly selected 5 seed sets of length 2 and evaluated against our systems, for which MAP scores were averaged. The result was most satisfying for the fruit category, reaching 0.810, 0.626, and 0.418 for its respective MAP measures. Results for the spice category were equal 0.630, 0.500, 0.360, while the meat category performed the weakest, reaching the scores 0.412, 0.3428, and 0.282.

7.3 Comparison with Previous Work

As established by Kushilevitz et al.(Kushilevitz et al., 2020), number of indicative patterns used (#patt), and number of candidate seed-term containing sentences (#sent) can be used for selecting and evaluating indicative patterns. Every value represents an average MAP on 5 seeds (chosen randomly, fixed for all values of #sent and #patt) of size 3. Our approach to the evaluation was similar; we have lowered the size of the seed sets from 3 to 2 to account for a smaller and more specific categories used.

According to Mamou et al., the performance of the algorithm was evaluated by the Mean Average Precision at different top n values (MAP@n). MAP@10, MAP@20 and MAP@50 on an English Wikipedia based dataset 11 are respectively 0.83, 0.74 and 0.63 (Mamou et al., 2018). These results appear to be somewhat higher than our final product, but do not considerably differ, offering a fair and satisfying comparison.

8 Discussion and Future Work

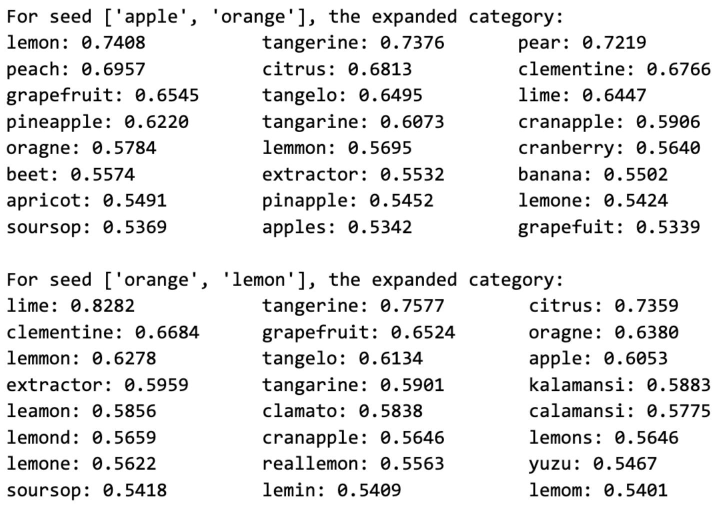
Given the size of our initial dataset and the cleverness of the implemented approach, we consider our system to be successful. After evaluating our approach, the strength of the design seems to be located in the specificity of the searched vocabulary, an excellent fit of the training data for this specific use, and finally good approaches to analyze short, independent, and instruction-like sentences. Although we are satisfied with the system we have designed, we came up with some potential changes that could improve its accuracy further.

8.1 Data Refinement

As one of the potential improvements, we could change or refine the dataset we have used to train our system. The current set of recipes, although it is strongly correlated to our task, also contains many spelling mistakes which made their way to the final results. Consequently, this may be lowering the accuracy of the words sets we are getting. In future, we may fix the errors in the dataset in advance, such as edit the spelling mistakes or set up root word sets. Besides, we will also look for other datasets that have cleaner data to further increase our system's accuracy.

8.2 Dependency Relations

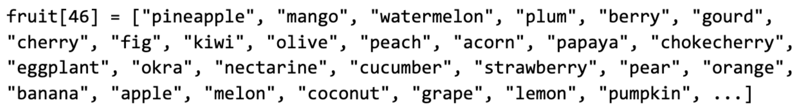
Additionally, we could also consider adding an additional context measurement element which could allow for further increase in accuracy of the results. For example, measuring the dependency relations between potential terms and other words present in the data set would be a great addition to the existing system. By including the type of the relation between the given word and the analyzed term (Mamou et al., 2018), we could we can create a more robust context metric that includes the directionality of the relationship rather



```
For seed ['apple', 'orange'], the expanded category:
lemon: 0.7408           tangerine: 0.7376       pear: 0.7219
peach: 0.6957           citrus: 0.6813          clementine: 0.6766
grapefruit: 0.6545      tangelo: 0.6495         lime: 0.6447
pineapple: 0.6220       tangarine: 0.6073       cranapple: 0.5906
oragne: 0.5784          lemmon: 0.5695          cranberry: 0.5640
beet: 0.5574            extractor: 0.5532       banana: 0.5502
apricot: 0.5491         pinapple: 0.5452        lemone: 0.5424
soursop: 0.5369         apples: 0.5342          grapefuit: 0.5339

For seed ['orange', 'lemon'], the expanded category:
lime: 0.8282            tangerine: 0.7577        citrus: 0.7359
clementine: 0.6684      grapefruit: 0.6524       oragne: 0.6380
lemmon: 0.6278          tangelo: 0.6134          apple: 0.6053
extractor: 0.5959       tangarine: 0.5901        kalamansi: 0.5883
leamon: 0.5856          clamato: 0.5838          calamansi: 0.5775
lemond: 0.5659          cranapple: 0.5646        lemons: 0.5646
lemone: 0.5622          reallemon: 0.5563        yuzu: 0.5467
soursop: 0.5418         lemin: 0.5409            lemom: 0.5401
```

Figure 6: The input and output of our system. By comparing the differences in responses from given seed sets, we can perceive the fluidity of the defined categories and the accuracy of estimating their semantic value.



```
fruit[46] = ["pineapple", "mango", "watermelon", "plum", "berry", "gourd",
"cherry", "fig", "kiwi", "olive", "peach", "acorn", "papaya", "chokecherry",
"eggplant", "okra", "nectarine", "cucumber", "strawberry", "pear", "orange",
"banana", "apple", "melon", "coconut", "grape", "lemon", "pumpkin", ...]
```

Figure 7: Fragment of a sample evaluation set representing the fruit category. Items in the set were obtained from a knowledge graph, representing an 'IsA' relation to the word 'fruit'.

that just a neighboring metric already provided by the Bag-of-Words measurement. Given the nature of our task, two main types of dependency relations might benefit our approach. First relation would include actions upon terms, which manifests itself through verb phrases associated with given terms (e.g., chop tomatoes, heat olive oil, garnish with cilantro). Another useful dependency would involve the analysis of metric units used in the kitchen, and closely associated with particular terms (e.g., cup of flour, pinch of salt, tablespoon of milk). These and other dependency relations can be found in cooking recipes, offering a great improvement to our solution.

Dependency relations method can also be considered as an alternative to bag-of-words method. For every parsing sentence, we can extract the dependency context for a specific words w with k modifiers and a header h. Therefore in the future work, we can explore further the dependency context method, as well as exploit the similarity and difference of these two methods to more fields.

9 Conclusion

We presented a working system for terminology set expansion, using two methodologies of context measurement. It equips the user with the ability to select a set of terms and expand it, making it a useful tool for writing, altering, or improving on cooking recipes. Such expanded term sets can then be used for many real-life situations and fulfill many time-consuming and manually-troubled requirements, confirming by providing our own examples. Considering the previous solutions to the problem which involved the definition of more generic categories, the specificity of our system seems to be an additional asset, improving on its functionality and testifying of the efficiency of such approach.

References

Gensim: Word2vec embeddings. https://radimrehurek.com/gensim/models/word2vec.html/. Accessed: 2021-11-11.

Natural language processing with python. https://www.nltk.org/book/. Accessed: 2021-11-11.

Recipenlg. https://recipenlg.cs.put.poznan.pl/. Accessed: 2021-11-11.

Damien Cram and Béatrice Daille. 2016. Terminology extraction with term variant detection. In *Proceedings of ACL-2016 System Demonstrations*, pages 13–18, Berlin, Germany. Association for Computational Linguistics.

Dmitry Davidov and Ari Rappoport. 2006. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 297–304, Sydney, Australia. Association for Computational Linguistics.

Guy Kushilevitz, Shaul Markovitch, and Yoav Goldberg. 2020. A two-stage masked LM method for term set expansion. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6829–6835, Online. Association for Computational Linguistics.

Jonathan Mamou, Oren Pereg, Moshe Wasserblat, Alon Eirew, Yael Green, Shira Guskin, Peter Izsak, and Daniel Korat. 2018. Term set expansion based NLP architect by Intel AI lab. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 19–24, Brussels, Belgium. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality.

Brian Roark and Eugene Charniak. 1998. Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. In *COLING 1998 Volume 2: The 17th International Conference on Computational Linguistics*.