Salt and Pepper Recipe for Ingredient Extraction

ABSTRACT

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

ACM proceedings, LaTeX, text tagging

1. INTRODUCTION

Transition from generic search to entity oriented search
What is entity oriented search, its importance. Instances.
why and how of recipe search. An example.?
entity in recipe-ingredient. Define?
Entity extraction in recipes poses several challenges. We discuss some of the interesting problems faced in this search vertical. The first step in this process is the crawling and classification of a recipe containing page. While there exist, several standard recipe formats on the web, such as the Recipe Markup Language (RecipeML), only a limited number of, usually large, websites conform to such standards. Thus, if only pages adhering to these identified as recipe containing pages, it leaves out several other unstructured recipes, particularly on the blogosphere. Even after correct identification of recipe pages, the isolation of actual entities such as ingredients is not a simple task. Even though, a static dictionary of ingredients seems to overcome this barrier, building such a dictionary is a daunting task. Moreover, new ingredients and their variations appear with time and a static dictionary is incapable of handling them. Further, the Ingredient Phrases, henceforth referred to as IP, are usually short and do not contain the contextual information which could be used to classify a token as part of a segment, or not.

There exists substantial literature on supervised techniques for entity extraction, where training is done on large annotated corpus and a rule based or statistical model is learnt. These methods work well for natural language text but are unsuited for cases where extraction is done on short phrases such as ingredient extraction from recipes and Named Entity Recognition (NER) in queries. Moreover, labeling is an expensive process where human expertise are required.

In the current work, we propose a completely unsupervised entity extraction technique that assumes only one constraint - candidate phrases from which extraction is to be done, must contain at least one entity. While, we chose ingredient extraction from recipes as our experimentation dataset, the technique can easily be extended to other applications and datasets.

The rest of the paper is organized as follows: Section 2 sheds light on the related work including the supervised and unsupervised techniques for entity extraction in existing literature. In Section 3 the dataset used in this work, is described. While Section 4 states our proposed algrithms and ideas, Section 5 details the evaluation criteria and the experiment results. Finally, Section 6 concludes the work, states our contribution and discusses future work.

2. RELATED WORK

Named Entity Recognition (NER), as a sub-field of Information Extraction (IE) has witnessed considerable attention in recent years because of the need to filter entities of interest from text. [9] provides a survey of traditional techniques in this domain. Techniques for NER fall under supervised machine learning[4], semi-supervised[5] and unsupervised[6] but these methods are suited for text documents, where a number of features are utilized for learning. However, they would not work well for scenarios where text segments are short and do not provide much contextual cues.

Recently, entity extraction from short text segments, such as web-queries[7, 11] has received attention.

recipe search [10]

Also related is the work on segmentation, which has been shown to improve retrieval performance in web-queries[3]. In this work, we use segmentation as a means to reduce the search space for entities and devise an unsupervised segmentation technique inspired from [8].

3. DATASET

The dataset for our experiments consists of 165 thousand recipes, crawled from 7 large, recipe websites. Each of these websites conformed to the Recipe Markup Language

WOODSTOCK '97 El Paso, Texas USA

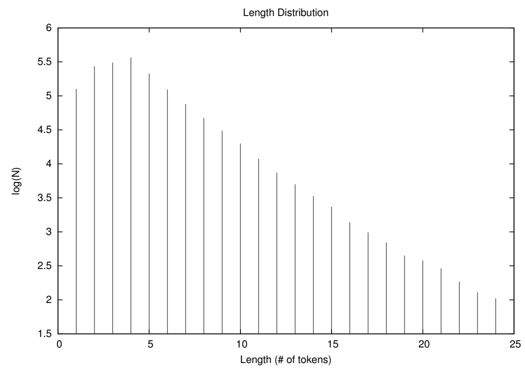

Figure 1: IP Length Distribution

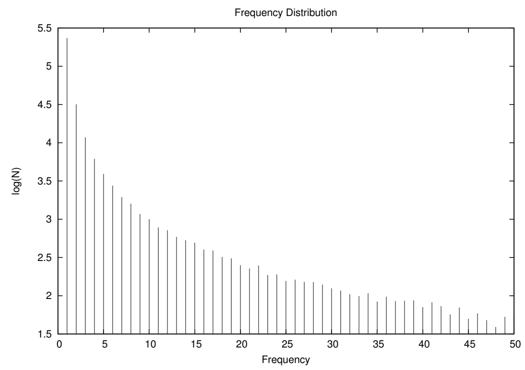

Figure 2: IP Frequency Distribution

(RecipeML) format and hence, each recipe was represented with fields such as title, nutrition-info, serving size, ingredients, preparations, etc. While, the recipe details are broadly organized under the mentioned attributes, certain fields - such as the ingredient field often contains extraneous information. Thus, extraction of base ingredients (represented in bold) from ingredient phrases (IP) such as '*1 1/2 cup fresh or frozen* **blueberri**, *divided*', ' *large* **shrimps** *in shell peeled deveined and cut crosswise into 0.5-inch pieces*' is important to restrict/expand the search results according to ingredient.

The corpus contains 1.6 million IP's (307 thousand distict IP's) with an average length of 4.16 tokens. The length distribution is shown in Fig.1. Unlike the length distribution of web-queries which follow a power law distribution [2], the IP's peak at length 5. For the current study, we focus on phrases with $l(IP) \leq 25$, ignoring higher lengths as they often contained spam or were the result of markup errors.

As depicted in Fig. 2, the IP frequency exhibits a power law distribution with a long tail, indicating the existence of few popular ingredients among the recipes while a large number of them have rare presence in recipes.

For evaluation, we annotated 600 IP's. These are sampled according to the frequency and length distribution of IP's. Labeling was a difficult task, since it was hard to state an unambiguous definition of ingredient.For example, in the phrase *superfine granulated sugar*, both 'sugar' and its super-set 'granulated sugar' could be termed as valid ingredients. It was even more difficult for instances such as *3 medium bananas, sliced and dipped in lemon juice*, where, while banana is the essential constituent, the ingredient is incomplete without presence of lemon juice.

We decided to label all the root ingredients of the phrase as true labels. Thus, in the above examples, 'sugar' and 'banana, lemon' were marked as ingredients. These scheme has obvious failings in terms of evaluation metrics but as we show in Section.5, our technique is capable of extracting the 'intuitive' ingredient with a high success rate.

4. OUR APPROACH

While, there exist several supervised approaches in literature, to extract entities, we decided to pursue an unsupervised technique since annotation is an expensive exercise. Moreover, in several scenarios, formulation of a precise definition of entity is difficult, if not impossible. We propose a simple, light-weight technique for entity extraction which does not require any manual intervention and uses only corpus statistics to identify entities.

As a pre-processing step, each IP was passed through the following filters:

- Each token which contained numerals was replaced by a dummy token 'NUM'. It resulted in all numerical values such as 7, 1/2, 3.5 being treated as the same.
- Mild stemming, to transform plurals - such as 'onions' to 'onion' was done. To minimize false positives such as 'boneless' from being stemmed, Part of Speech (POS) tagging was done and only Noun Phrases (NP) were considered as candidates for stemming.
- All punctuations, except '(', ')', ',', '-' were removed. This was done to prevent these characters from causing statistical variations in the corpus. While '-' was retained, as it is, the other 3 characters were used as markers to indicate user-defined segment boundaries.

The technique proposed in this work is based on two simple observations:

1. Each Ingredient Phrase (IP) contains at least 1 ingredient.
2. Likelihood of an entire IP, to be an ingredient decreases with increasing length (in terms of tokens/segments) of the phrase.

Taking these two observations as the basis we propose 3 extraction formulations which are variations of each other. Each these takes the list of IP's as input. Formulation 2 & 3 also take a corpus generated Stop Words List (SWL) as input. A frequency dependent threshold $\tau(l(ip))$ is used as the selection criteria for qualification to the Candidate Ingredients List (CIL), which is returned as output. This CIL, is used in a maximal search extraction (discussed later), to predict the ingredients present in an IP. The threshold is inspired from observation 2, stated above, and is exponential in the length of IP. Thus, while single-word IP's required a small frequency occurance, this frequency requirement increased exponentially with increasing length. The SWL was constructed by merging a list of standard English stop-words - INQUERY [1] with corpus dependent stop-words, to produce the list. The corpus dependent list was produced by selecting 200 terms with the highest term frequency among the IP's. This was further, manually perused to remove the false positives, resulting in a list of 551 stop-words.

The first formulation, stated in 1 simply selects the IP's which satisfy the qualification criteria and adds them to the CIL.

In Formulation 2, stop-words were removed from IP's using the SWL. The rationale behind this variation was that several IP's which failed to qualify for the CIL, could manage to do so after SW removal, due to reduced lengths.

```
Algorithm 1: Formulation 1
input  : List of IP
output: Candidate Ingredients
$CIL \leftarrow \emptyset$
for $ip \in IP$ do
    if $f(ip) \geq \tau(l(ip))$ then
        $CIL \leftarrow ip$
return $CIL$
```

The 3^{rd} formulation, is an iterative variation of the 2^{nd} where during each iteration, besides populating the CIL, the SWL is also enhanced by appending the $top-k$ tokens with highest term frequency.

```
Algorithm 2: Formulation 2
input  : List of IP, List of SW
output: Candidate Ingredients
$CIL \leftarrow \emptyset$
for $ip \in IP$ do
    foreach $token \in ip$ do
        if $token \in SW$ then
            Remove $token$ from $ip$
for $ip \in IP$ do
    if $f(ip) \geq \tau(l(ip))$ then
        $CIL \leftarrow ip$
return $CIL$
```

We also explore the technique of unsupervised segmentation of phrases to improve the extraction performance. The motivation, for using segmentation is that it helps to separate the terms in an IP into segments such that each segment maps to a semantic component or concept. The segment boundaries can help in separation of ingredient segments from non-ingredient segments, reducing the search space for ingredients.

A variation of the Pointwise Mutual Information (PMI) score was used as the basis for segmentation. For two n-grams x and y, PMI is defined as:

$$PMI(x;y) = \log \frac{p(x,y)}{p(x)p(y)} \tag{1}$$

where $p(x,y)$ is the joint probability of x and y. This can be computed as $p(x,y) = f(x \wedge y)/N$, with the numerator representing the number of IP's containing both x and y and N is the total number of IP's. $p(x)$ and $p(y)$ represent the marginal probabilities of these grams and amount to $p(x) = f(x)/N$ and $p(y) = f(y)/N$. Since, computing the PMI of all n-grams is unnecessary (we maintain only the top-k PMI score n-grams) and time-consuming, we use Algorithm. 4 to produce the highest scoring grams. This is an iterative algorithm, that generates bi-grams at each iteration and subsequently merges the top-k grams with highest PMI score. PMI suffers from the shortcoming that it only captures the co-occurance, while ignoring the actual number of occurances (frequency) of the grams. Thus, two grams which have only single occurances in the log, possibly due to a misspelling, and this occurance is in a single IP, will have a high PMI score. To incoporate this element of frequency in our formulation, we used $\log(f_{x,y}) * PMI$ as our scoring function, instead of PMI.

```
Algorithm 3: Formulation 3
input  : List of IP, List of SW
output: Candidate Ingredients
$CIL \leftarrow \emptyset$
$SWL \leftarrow$ List of SW
for $i \leftarrow 1$to $numIterations$ do
    for $ip \in IP$ do
        foreach $token \in ip$ do
            if $token \in SWL$ then
                Remove $token$ from $ip$
    for $ip \in IP$ do
        if $f(ip) \geq \tau(l(ip))$ then
            $CIL \leftarrow ip$
    $DF_{tokens} \leftarrow (token \in IP, \mathrm{DF}(token))$
    $top-k_{DF} \leftarrow$ top-k DF tokens $\in DF_{tokens}$
    for $token \in top-k_{DF}$ do
        $SWL \leftarrow token$
return $CIL$
```

```
Algorithm 4: PMI Score Computation Algorithm
input  : List of IP
output: PMI Scores of n-grams
$gramHash \leftarrow \emptyset$
for $ngram \leftarrow 2$ to $maxNGram$ do
    for $ip \in IP$ do
        $bigramSet \leftarrow$ GenBiGrams($ip$)
        for $bigram \in bigramSet \wedge bigram \notin gramHash$ do
            $gramHash \leftarrow (biGram$, PMI($biGram$))
    for $top-k$ $PMI(biGram)$ do
        DoMerge($biGram$)
return $gramHash$
```

Having computed the scores, we experimented with 3 unsupervised segmentation techniques on each IP.

1. Bigram Segmentation: The IP is traversed from left to right and for each bigram, if the PMI score of bigram is less than a pre-specified threshold, a segment boundary is introduced.

2. Linear n-gram Segmentation: This a variation of the previous segmentation technique such that traversal is done from left to right. But here, the candidate grams

Formulation	Freq F-Score
1	0.421
2	0.720
3	0.722

Table 1: IE without Segmentation

can be greater than 2-grams and a segment boundary is introduced only if the PMI score of candidate is less than the threshold.

3. Globally Optimal Segmentation: A score for each possible segmentation of IP is computed by adding the PMI scores of individual segments. The segmentation that yields the highest score is given as output. A dynamic programming approach was used to search over all possible segmentations.

The segmentation variations of the 3 formulations, treats each segment as an indivisible unit, i.e. instead of tokens, the segments are treated as the unit of processing.

Once, the CIL is built, ingredient extraction from an IP involves maximal search of all n-grams/n-segments in CIL. All n-grams/n-segments found in CIL are predicted as ingredients. If none is found, all n-grams/n-segments are predicted as ingredients. Maximal search implies that only such n-grams/n-segments are predicted as ingredients where $n - gram/n - segment \in CIL$, and no *(n+1)-gram/(n+1)-segment* $\in$ *CIL,* $\forall$ *(n+1)-gram/(n+1)-segment* $\supset$ *n-gram/n-segment.*

5. EXPERIMENTAL RESULTS

Experiments were done to validate our claims and to identify the best performaning technique and parameters. As mentioned in Section 3 600 IP's were labeled and performance evaluation was done on labeled ingredients in these IP's. We used a variation of F-score as the evaluation metric. F-score is the harmonic mean of precision and recall and can thus be computed as:

$$F = \frac{2 \times precision \times recall}{precision + recall} \qquad (2)$$

For the current system we used a frequency multiplied F-score to incorporate the frequency of IP's from which extraction was done. It can be formulated as:

$$Freq - F = \frac{\sum_{i=1}^{600}(f_i \times F_i)}{\sum_{i=1}^{600} f_i} \qquad (3)$$

In effect, this alteration results in contribution proportionate to the frequency of occurance of IP in the log and is a better reflection of our techniques performance, than the original F-score.

The first set of experiments on IP's after initial pre-processing are shown in 5. These experiments treated tokens as the unit of processing and the use of a stop-words list did result in improvements, as indicated by the change of Freq F-Score in Formulation 2 and 3.

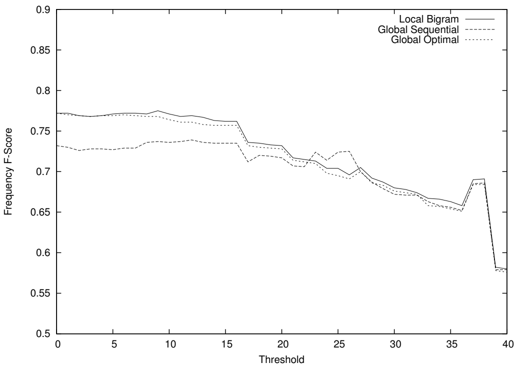

Figure 3: Effect of (Segmentation) Threshold on Formulation 3

Segmentation Technique	Formulation	Freq F-Score
Local Bigram	1	0.651
Local Bigram	2	0.769
Local Bigram	3	0.771
Global Sequential	1	0.675
Global Sequential	2	0.732
Global Sequential	3	0.736
Global Optimal	1	0.464
Global Optimal	2	0.768
Global Optimal	3	0.772

Table 2: IE with Segmentation

The second set of experiments were conducted by applying segmentation on the IP's, as described in Section 4. This involved setting a cutoff threshold for qualifying to the Candidate Ingredients List (CIL). Figure 3 shows the change in performance corresponding to threshold. The drop in performance with increasing threshold can be attributed increased splitting of ingredients due to higher thresholds.

Table 5 depicts the effect of segmentation on all the 3 formulations and it can be observed that segmentation results in improvements in each one of them.

It can be seen that even with our light-weight technique reasonable Frequency F-score performance could be achieved. Moreover, we argue that while numerical metrics like Frequency F-score are needed for completeness of evaluation, they are inept at capturing the overall performance of the technique. Table 3 gives a few examples of ingredient extraction using our technique. Set 1 shows instances where the technique performed perfectly and the extracted ingredients matched the annotations, while set 2 shows examples where, even though the claimed ingredients did not match the annotations, they are not absolutely incorrect as the evaluation metric would treat it. Set 3 shows a couple of examples where the system fails completely in identifying the ingredients.

6. CONCLUSION

7. REFERENCES

[1] J. Allan, M. Connell, W. B. Croft, F. Feng, D. Fisher, and X. Li. Inquery and trec-9. In *Proceedings of TREC-9*, pages 551–577, 2000.

[2] M. Bendersky and W. B. Croft. Analysis of long queries in a large scale search log. In *Proceedings of the 2009 workshop on Web Search Click Data*, WSCD '09, pages 8–14, New York, NY, USA, 2009. ACM.

Original IP	Annotated Ingr	Predicted Ingr	Predicted Ingr (segmentation)
1 teaspoon kosher salt	kosher salt	kosher salt	kosher salt
firm ripe bartlett or anjou pear	bartlett, anjou pear	anjou pear	anjou pear
0.25-cup reduced-sodium soy sauce	soy sauce	reduced-sodium soy sauce	soy sauce
1 pound yukon gold potatoe	potatoe	yukon gold potatoe	potatoe
0.5 teaspoon dry mustard	mustard	dry mustard	dry mustard
chopped napa chinese cabbage	napa chinese cabbage	cabbage	cabbage
dill cucumber pickle	dill cucumber pickle	dill, cucumber, pickle	dill, cucumber, pickle
additional cornmeal for dusting	cornmeal	additional, cornmeal	additional, cornmeal

Table 3: Extraction performance on Ingredient Phrases

[3] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *EMNLP-CoNLL'07*, pages 819–826, 2007.

[4] A. E. Borthwick. *A maximum entropy approach to named entity recognition.* PhD thesis, New York, NY, USA, 1999. AAI9945252.

[5] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *In Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 100–110, 1999.

[6] O. Etzioni, M. Cafarella, D. Downey, A. maria Popescu, T. Shaked, S. Soderl, D. S. Weld, and E. Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165:91–134, 2005.

[7] J. Guo, G. Xu, X. Cheng, and H. Li. Named entity recognition in query. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '09, pages 267–274, New York, NY, USA, 2009. ACM.

[8] N. Mishra, R. Saha Roy, N. Ganguly, S. Laxman, and M. Choudhury. Unsupervised query segmentation using only query logs. In *Proceedings of the 20th international conference companion on World wide web*, WWW '11, pages 91–92, New York, NY, USA, 2011. ACM.

[9] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26, January 2007. Publisher: John Benjamins Publishing Company.

[10] S. Oyama, T. Kokubo, and T. Ishida. Domain-specific web search with keyword spices. *IEEE Transactions on Knowledge and Data Engineering*, 16:17–27, 2004.

[11] M. Paşca. Weakly-supervised discovery of named entities using web search queries. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 683–690, New York, NY, USA, 2007. ACM.