# A semi-supervised approach to extracting multiword entity names from user reviews

Olga Vechtomova
Department of Management Sciences, Faculty of Engineering
University of Waterloo, Waterloo, ON, Canada
ovechtom@uwaterloo.ca

## ABSTRACT
The paper describes a semi-supervised approach to extracting multiword units that belong to a specific semantic class of entities. The approach uses a small set of seed words representing the target class, and calculates distributional similarity between the candidate and seed words. We adapt a well-known document ranking function, BM25, to the task of calculating similarity between vectors of context features representing seed words and candidate words, and perform a systematic comparison to a number of distributional similarity measures. We then introduce a method for ranking multiword units by the likelihood of belonging to the target semantic class. The task used for evaluation is extraction of restaurant dish names from the corpus of 157,865 restaurant reviews.

## Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Retrieval models.

## General Terms
Algorithms, Design, Experimentation.

## Keywords
Information extraction, entity retrieval

## 1. INTRODUCTION
There is a growing number of applications that require identification of entities of a certain semantic class in specialised corpora. For instance, recommender systems that give advice to users on businesses or products based on user reviews could benefit from methods that extract specific aspects of products/businesses from user reviews. An example of a specific application is a restaurant recommendation system that lists the names of dishes served in the restaurants that it recommends. Manually created knowledge bases are often inadequate, as they may not have sufficient coverage of specialised lexicons, and require substantial human effort to build and keep up to date.

In this paper we describe a semi-supervised approach to extracting entities (both single words and multiword units) of a specific semantic class from user-written reviews. The method starts with a small initial set of seed words, and uses distributional similarity measures to rank all single words in the corpus by similarity to the seeds. The similarity is calculated between the words' feature vectors, where the features of a word are the grammatical dependency triples it co-occurs with. An example of a dependency triple is "eat V:obj:N pizza". We compare a number of distributional similarity measures, namely Lin's measure [1], a measure by Weeds and Weir [2] and a directional similarity measure balAPinc by Kotlerman et al. [3]. One of the novel contributions of this work is the adapting of a document retrieval model BM25 by Robertson et al. [4] to act as a term-term distributional similarity measure.

After ranking single words by similarity to seed words, we combine their scores to rank multiword units (MWUs). We evaluate a number of methods of scoring multiword units based on the scores of single words. The evaluation is done on a corpus of 157,865 restaurant reviews. The task consists of extracting dish names from restaurant reviews, which could be either single nouns or MWUs. This task is complicated by the fact that reviewers often use subjective modifiers in front of dish names, therefore part of a challenge is separating the dish name proper from such modifiers. As part of our method we extract subjective adjectives using the same weakly supervised approach, and remove them from the MWU dish names.

## 2. RELATED WORK
There exist a number of corpus-based methods for extracting words belonging to the same semantic class. Based on the amount of training data required, there are three categories of methods: unsupervised, semi-supervised and supervised. Tsai and Chou [5] proposed an unsupervised method for the extraction of dish names from Chinese blogs using suffix arrays and conditional random fields. Supervised methods typically use classifiers trained on a large amount of manually annotated data, e.g., [6]. Semi-supervised methods require only a small amount of training data, usually in the form of seed words. These methods are generally more attractive for practical applications than supervised methods since less manual labour is needed. Examples of semi-supervised methods are Meta-Bootstrapping [7], Basilisk [8] and Snowball [9]. Semi-supervised approaches rely on either (a) a number of hand-crafted extraction patterns, or (b) co-occurrence information of lexical units, or (c) their distributional similarity. A comprehensive review of co-occurrence-based and distributional similarity approaches is given in [10].

One of the earliest methods using extraction patterns is a hyponymy detection method by Hearst [11], who uses six patterns (e.g. "such *X* as *") and a category name *X* to identify its hyponyms. Lately, a number of other methods using patterns were proposed, e.g. [12-14]. Wang and Cohen [12] apply Hearst's hyponymy patterns to get an initial set, and then expand it by a web-based algorithm from the pages containing seeds. Etzioni et al. [13] also use a set of hyponymy patterns, and select candidates using Pointwise Mutual Information (PMI). Kozareva et al. [14] use a doubly-anchored pattern "*X* such as *Y* and *" and a bootstrapping algorithm. All these methods require the scale and redundancy of the web, as the extraction patterns may not be frequently observed in smaller corpora.

Co-occurrence based methods rely on such statistical measures as PMI [15], $\chi^2$ (Chi-square) [16] and Log-likelihood ratio [17]. Riloff and Shepherd [18] rely on the co-occurrence of nouns in small window sizes with a small set of seeds. Yarowsky [19] uses co-occurrence of words in windows of 100 words with words from a specific Roget thesaurus category to identify lists of words

salient to this category. One limitation of the co-occurrence based measures is that words have to occur in the vicinity of the known words in order to be extracted. This limitation is overcome by distributional similarity methods.

According to the distributional similarity principle, words that occur in similar contexts are likely to have similar meanings. A number of symmetric and asymmetric distributional similarity measures have been proposed. Measures proposed by Lin [1] and Weeds and Weir [2] are the most well-known symmetric distributional similarity measures. Lin's measure [1] uses grammatical dependency relations as features, weighted using Mutual Information. Weeds and Weir [2] describe a general framework for computing distributional similarity measure based on the concepts of precision and recall. Kotlerman et al. [3] propose an asymmetric (directional) similarity measure, balAPinc, designed to find words with more specialised meaning compared to the seed. They adapt the concept of average precision from Information Retrieval to calculate similarity between two words. In their approach the features in the vector of the seed word are analogous to the complete set of relevant documents, while the features in the vector of the candidate word are analogous to the retrieved documents.

Distributional similarity methods also differ by the linguistic units they use as context. For example, Pantel et al. [20] use noun phrase chunks to the left and right of a term as its context, while Lin [1] and Kotlerman et al. [3] use grammatical dependency relations as features. Weeds and Weir [2] use verbs as features when calculating similarity between nouns. As discussed in [21], the use of grammatical relations as features leads to the identification of "tighter" relationships between words, whereas the use of document- and sentence-level word co-occurrences would lead to the identification of "looser" relationships. So, while the former are better for identifying words belonging to the same semantic class, the latter are more appropriate for grouping words into subject categories. In our approach we use the former.

To our knowledge document ranking models have not been previously evaluated on the task of calculating term-term distributional similarity. In this paper we evaluate a well-known document ranking function BM25 [4] on the task of extracting restaurant dish names and subjective adjectives and compare it with three state-of-the-art distributional similarity metrics. The reason for selecting BM25 is that (a) it is one of the best performing IR functions, and (b) it has tuning constants that can moderate the effect of document length and term frequencies. When applied to the task of computing distributional similarity between terms, this means that we can adjust the effects of feature vector length and feature frequencies. One of the goals of the evaluation experiment is to compare BM25 to Lin's [1], Weeds and Weir's [2] and Kotlerman's [3] distributional similarity measures. Comparison to methods using extraction patterns or co-occurrence based measures is out of the scope of this paper.

# 3. RANKING SINGLE WORDS BY DISTRIBUTIONAL SIMILARITY TO SEEDS

We start the process with a small set of seeds, which in our task are single nouns denoting dish names, and a set of words we want to rank (candidate words), that comprise all single nouns in the corpus. In this section we describe the process of building feature vectors for both seeds and candidate words, and the process of ranking candidates with respect to seeds by using the adapted BM25 function.

## 3.1 Building feature vectors

The context of each seed and candidate word is represented as a vector of context features. As a context feature we use grammatical dependency triples, in a similar way as was done by Lin [1]. In detail, our method consists of the following steps:

1. Perform dependency parsing[1] of the corpus. Each dependency triple consists of two words, their POS tag and a dependency relationship that connects them, for example: "eat VB:dobj:NN pasta".

2. For each seed word *s*, extract all dependency triples that contain it, and build a vector feature from each triple, by substituting the word *s* with "X". For instance, if we build a vector for the word "pasta", the dependency triple "eat VB:dobj:NN *pasta*" is transformed into: "eat VB:dobj:NN *X*". In our method we introduce a tuning constant *t*, which represents the number of seed words that a feature has to co-occur with in order to be included in the vector. The set $\{F\}$ contains all features that have the seed co-occurrence frequency greater than *t*. Only these features are included in the vectors of seed words.

3. For each candidate word *c* extract all dependency triples with which it co-occurs, transform them into features in the same way as in Step 2. Include in the vector only the features that belong to set $\{F\}$.

4. For each feature in the vector of a seed or candidate word, record *TF*, which is the frequency of co-occurrence of the feature with this word in the corpus. In the example above, *TF* of the feature "eat VB:dobj:NN *X*" in the vector of the word "pasta" is the frequency of occurrence of "eat VB:dobj:NN *pasta*" dependency triple in the corpus. We used the parsed corpus of 157,865 restaurant reviews to get frequencies of triples.

## 3.2 Computing similarity between vectors

The objective is to rank all candidate words in the corpus by similarity to all seed words. The first step is to calculate similarity in a pairwise manner between each seed and each candidate word. The result of this step is a ranked list of candidates for each seed. The second step is combining these lists into one ranked list.

In order to compute similarity between vectors of a seed and a candidate, we adapt BM25 document ranking model with query weights, called Query Adjusted Combined Weight (QACW) [23]. In the QACW formula, the vector of the seed word *s* is treated as the query and the vector of the candidate word *c* as the document:

$$QACW_{c,s} = \sum_{f=1}^{F} \frac{TF(k_1+1)}{K+TF} \times QTF \times IDF_f \quad (1)$$

Where: *F* – the number of features that a candidate word *c* and a seed word *s* have in common; *TF* – frequency of feature *f* in the vector of candidate word; *QTF* – frequency of feature *f* in the vector of the seed entity (computed in the same way as *TF*); $K = k_1 \times ((1-b)+b \times DL/AVDL)$; $k_1$ – feature frequency normalisation factor; *b* – document length normalisation factor; *DL* – number of features in the vector of the candidate entity; *AVDL* – average number of features in all candidate entities. The *IDF* (Inverse Document Frequency) of the feature is calculated as follows:

$$IDF_f = log\frac{N}{n_f} \quad (2)$$

[1] We used Stanford dependency parser [22].

Where, $n_f$ – number of candidate word vectors the feature *f* occurs in, *N* – number of candidate word vectors.

After all candidate words are ranked by similarity to each seed, their scores in each ranked list are normalised so that they are in the range between zero and one. The normalised scores of the candidate word *c* in the ranked lists for all seed words are then summed:

$$SeedBM25_c = \sum_{s=1}^{S} QACW_{c,s} \quad (3)$$

# 4. EXTRACTING AND RANKING MULTIWORD UNITS

Here we describe a method for extracting and ranking nominal MWUs by the likelihood of belonging to the target semantic class. After ranking all single nouns[2] in the corpus by similarity to all seeds, as described in Section 3, we take all noun phrases (NPs) output by an NP chunker, remove subjective adjectives from them, and rank them based on the seed-similarity scores of the nouns they contain.

## 4.1 Removing subjective adjectives

Noun phrases output by NP-chunkers may contain subjective adjectives that are not part of the MWU proper, for example "delicious" in "delicious Italian pizza" is not part of the dish name "Italian pizza". Such adjectives have to be identified and removed. We apply the same weakly supervised method as described in Section 3 for identifying subjective adjectives. All adjectives in the corpus are extracted and ranked by similarity to a set of seeds. We then take *a* top ranked adjectives and treat them as subjective adjectives in the following algorithm: (1) In each NP, find the rightmost occurrence of a subjective adjective; (2) Remove words preceding (and including) this adjective in the NP. The motivation for doing so is that a subjective modifier typically occurs early in an NP, and usually anything that precedes it is not part of the entity name, for instance "their delicious Italian pizza".

## 4.2 Ranking noun phrases

The next step is to rank all noun phrases output in the previous stage, based on the seed-similarity scores of the single nouns they contain. In developing the NP ranking function, we were guided by an intuition that the further away the noun is from the head of the NP, which is commonly assumed to be the rightmost word, the less its score should contribute to the overall score of the NP. For instance, if our task is to find dish names, and we obtain a score of 0.5 for "pizza" and 0.2 for "restaurant", intuitively "restaurant pizza" should be weighted higher than "pizza restaurant". To achieve this we propose to discount noun scores based on the distance from the end of the NP. We evaluate the following discount factors:

**Table 1. Discount factors.**

| | |
|---|---|
| Log-linear | $D = 1 - log_{10}(d_i)$ |
| Linear | $D = 1 - ((d_i - 1) \times 0.1)$ |
| No discount | $D = 1$ |
| 0.5 discount | $D = \begin{cases} 1 & if\ d_i = 1 \\ 0.5 & otherwise \end{cases}$ |

[2] POS tagging and NP chunking were done using OpenNLP.

Here $d_i$ is the distance of the noun *i* from the end of the NP, with *d* of the last word being 1.

$$NPscore = \frac{\sum_i^n Dw_i}{n} \quad (4)$$

Where: $w_i$ – seed-similarity score of the noun *i* calculated according to Eq. 3 or using any of the other three similarity measures that we evaluate in our experiments; *D* – discount function, *n* – number of words in the NP. Linear discount function showed better performance compared to others, therefore it is used in our experiments. In Section 6.2, we report performance comparison of these discount functions.

# 5. Evaluation

The evaluation goals are as follows:

1. Evaluate the effectiveness of ranking single words by similarity to the set of seed words, using (a) the BM25-based measure, (b) Lin's measure, (c) Weeds and Weir measure and (d) balAPinc measure. Evaluation is done on two tasks:

- extraction and ranking of single-noun dish/food names
- extraction and ranking of subjective adjectives

2. Evaluate the method for extracting and ranking multiword units as members of a specific semantic class. The task consists of identifying names of restaurant dishes, many of which are MWUs.

## 5.1 Dataset

The dataset used for evaluation consists of 157,865 restaurant reviews from CityGrid. We randomly selected 600 restaurant reviews, where all dish names and their subjective modifiers were manually labeled by two annotators. Each annotator labeled 300 reviews, then the third person acted as adjudicator, going through all labeled dish names and correcting errors. In total, 1000 multiword and single-noun distinct dish names were labeled by the annotators in 600 reviews. The annotators labeled different sets of reviews, therefore agreement cannot be calculated. However, prior to this, the annotators were asked to label dish names in the same set of 50 reviews. Annotator A labeled 156 dish names, and annotator B labeled 143 dish names. They agreed on 105 dish names, which is a reasonable level of agreement.

For the purpose of evaluating the first stage of the method – single word ranking – all single nouns were extracted from the 1000 dish names, and the annotators went through them selecting those nouns that refer to food. The reason why we did not automatically use all nouns is that not all nouns in a MWU dish name are referring to food, for example, "field" in "field salad". In this manner we identified 573 unique single-word food/dish names.

To obtain a set of subjective adjectives, all adjectives were extracted from subjective modifiers labeled by annotators, giving us 472 unique subjective adjectives.

## 5.2 Evaluation of single word extraction

In this section we describe the evaluation of the BM25 based method (Section 3) and three distributional similarity methods on the ranking of single words: (1) dish names and (2) subjective adjectives.

### *5.2.1 Dish name extraction and ranking*

We generated 20 seed sets, each consisting of 10 seed words. Sets 1-10 were used for tuning the system parameters, while sets 11-20 were used for testing. Seed sets were generated as follows: 573 single-word food names were ranked by frequency of occurrence in the 600 reviews, then 20 seed sets were generated randomly from the list of 100 top-ranked nouns.

The seed-threshold tuning parameter ($t$) was evaluated in the range [1-7] for all measures. The parameters $b$ and $k_1$ in BM25 were evaluated with the values in the range [0.1-0.9] and [0.2-1.6] respectively. The best results were obtained with $b$=0.9 and $k_1$=1.6. Weeds' method has two tuning parameters: $\beta$ and $\gamma$ [2], which performed best with values 1 and 0.8 on the training set.

Candidate words include all single nouns (3,271) extracted from 600 reviews. The measures used for evaluation are Mean Average Precision (MAP) and Precision at different cut-offs (at 50, 100 and 200 ranked words). Results are presented in Table 2 for the training sets and in Table 3 for the test sets. In all the following tables we indicate statistical significance of BM25 runs compared to the distributional similarity measure that showed the best MAP in that table. Statistical tests were conducted using paired t-test (* means p<0.05, while ** means p<0.01). In both training and test seed sets, the best MAP was achieved by balAPinc measure. Precision at 50, however, was better with BM25, showing statistically significant improvement over balAPinc on the training set.

**Table 2. Results for the noun training seed sets (1-10).**

| Run | MAP | P@50 | P@100 | P@200 |
|---|---|---|---|---|
| Lin (t=3) | 0.5302 | 0.956 | 0.882 | 0.7685 |
| Weeds (t=1) | 0.5533 | 0.854 | 0.804 | 0.7375 |
| balAPinc (t=2) | 0.5797 | 0.942 | 0.889 | 0.8065 |
| BM25 (t=1) | 0.5791 | 0.98** | 0.908 | 0.829 |

**Table 3. Results for the noun test seed sets (11-20).**

| Run | MAP | P@50 | P@100 | P@200 |
|---|---|---|---|---|
| Lin (t=3) | 0.5255 | 0.968 | 0.893 | 0.7755 |
| Weeds (t=1) | 0.5501 | 0.886 | 0.795 | 0.7445 |
| balAPinc (t=2) | 0.5836 | 0.964 | 0.91 | 0.82 |
| BM25 (t=1) | 0.5705 | 0.97 | 0.893 | 0.811 |

In both the training and test seed sets the best MAP was achieved by balAPinc measure. Precision at 50, however, was better with BM25, showing statistically significant improvement over balAPinc on the training set.

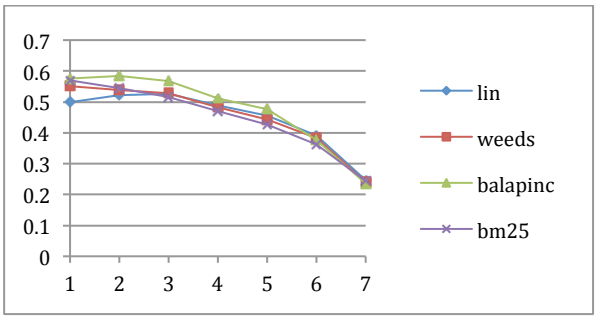


**Figure 1. Effect of seed threshold on MAP (noun ranking).**

The best MAP of all runs on the test set was achieved with the same seed-threshold parameter values as on the training sets. Figure 1 shows the effect of seed-threshold on MAP based on the test sets results (Table 3). A possible explanation of why BM25-based measure works best with seed threshold ($t$) set to 1, is that smaller $t$ leads to larger number of features in the vectors of seeds. In our approach, the vector of a seed is analogous to a query in document retrieval, therefore, a larger number of features in the vector of a seed is similar to having a longer query in document retrieval. Generally, longer queries are associated with higher performance of IR models, such as BM25. Hence, it is expected that BM25 would also work better with longer seed vectors.

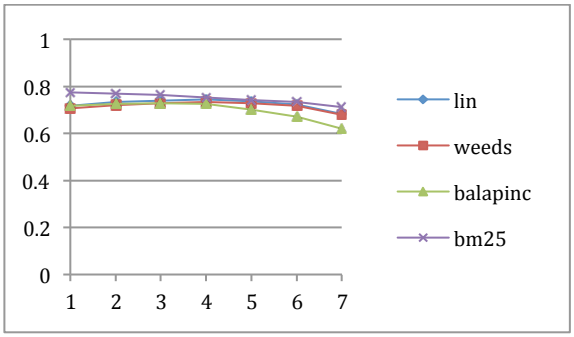


**Figure 2. Effect of seed-threshold on MAP (adjective ranking).**

### *5.2.2 Subjective adjective extraction.*

Twenty seed sets, each consisting of 10 seed words, were randomly generated from the 100 top-frequent subjective adjectives using the same method as was used for dish name seed set generation. Sets 1-10 were used for training the system parameters, while sets 11-20 were used for testing. Results on the training and test sets are given in Tables 4 and 5 respectively.

**Table 4. Results for the adjective training seed sets (1-10).**

| Run | MAP | P@50 | P@100 | P@200 |
|---|---|---|---|---|
| Lin (t=4) | 0.7355 | 0.916 | 0.861 | 0.83 |
| Weeds (t=4) | 0.7225 | 0.902 | 0.857 | 0.8205 |
| balAPinc (t=3) | 0.7143 | 0.858 | 0.839 | 0.797 |
| BM25 (t=1) | 0.7654** | 0.902 | 0.878* | 0.848 |

**Table 5. Results for the adjective test seed sets (11-20).**

| Run | MAP | P@50 | P@100 | P@200 |
|---|---|---|---|---|
| Lin (t=4) | 0.7442 | 0.914 | 0.883 | 0.842 |
| Weeds (t=4) | 0.7334 | 0.916 | 0.878 | 0.835 |
| balAPinc (t=3) | 0.7296 | 0.892 | 0.859 | 0.812 |
| BM25 (t=1) | 0.7744** | 0.922 | 0.889 | 0.861 |

BM25 showed statistically significant improvement in MAP compared to all three other measures (p<0.01) on both training and test sets. Improvements in Precision at cut-offs over Lin's measure, which has the highest MAP, are significant only on the training set at 100.

## 5.3 Evaluation of MWU dish name extraction and ranking

As input to this stage we use all (5,257) distinct noun phrases (NPs) extracted by the NP-chunker from the 600 restaurant reviews. We then proceed to perform two steps as described in Section 4. Step 1: Remove subjective adjectives from the NPs. Step 2: Rank NPs based on the seed-similarity scores of their constituent single nouns. The scores are obtained using one of the four evaluated similarity measures.

Step 1 was performed by removing top $a$ adjectives ranked by one of the four similarity measures. We evaluated values of $a$ from 0 to 400 in the increments of 50. Zero here means that no adjectives

were removed. Figure 3 shows the effect of *a* on performance. We also evaluated removal of all adjectives, marked "all" in Figure 3.

Step 2 was performed by using all single nouns ranked by one of the four similarity measures. In each run we used the same similarity measure for both Step 1 and Step 2, for instance, "Lin (a=200)" run denotes that Lin's measure was used to rank adjectives, top 200 of which were removed from NPs, then single nouns, also ranked by Lin's method were used for scoring the resulting NPs. For each run we used parameter values that showed the best performance on the training sets described in the previous section. For instance, for Lin's method, we used *t*=3 for ranking nouns, and *t*=4 for ranking adjectives.

**Table 6. MWU ranking results for the training sets.**

| Run | MAP | P@50 | P@100 | P@200 |
|---|---|---|---|---|
| Lin (a=50) | 0.3747 | 0.93 | 0.824 | 0.7595 |
| Weeds (a=50) | 0.3524 | 0.876 | 0.787 | 0.706 |
| balAPinc (a=50) | 0.3751 | 0.884 | 0.812 | 0.7215 |
| BM25 (a=100) | 0.3858* | 0.852 | 0.794 | 0.722 |

**Table 7. MWU ranking results for the test sets.**

| Run | MAP | P@50 | P@100 | P@200 |
|---|---|---|---|---|
| Lin (a=50) | 0.3738 | 0.92 | 0.831 | 0.759 |
| Weeds (a=50) | 0.3483 | 0.854 | 0.787 | 0.684 |
| balAPinc (a=50) | 0.3742 | 0.886 | 0.814 | 0.7245 |
| BM25 (a=100) | 0.3814 | 0.832 | 0.779 | 0.715 |

BM25 has the highest MAP compared to the other three measures. The improvement on the training set over balAPinc, which has second highest MAP, is statistically significant ($p<0.03$). However, balAPinc gives somewhat better precision at 50, 100 on both training and test sets, and at 200 on the test set.

The best results for BM25-based similarity measure were obtained with *a*=100, and for Lin's, Weeds' and balAPinc with *a*=50. Comparison of different values of *a* (Figure 3) shows that removal of top-ranked subjective adjectives helps performance, for instance performance with up to 150 top adjectives removed is statistically better ($p < 0.05$) than no adjective removal with all four similarity measures. Removal of more adjectives than this tends to degrade performance, since more adjectives that are legitimate parts of dish names are removed, such as "white" in "white chocolate mousse".

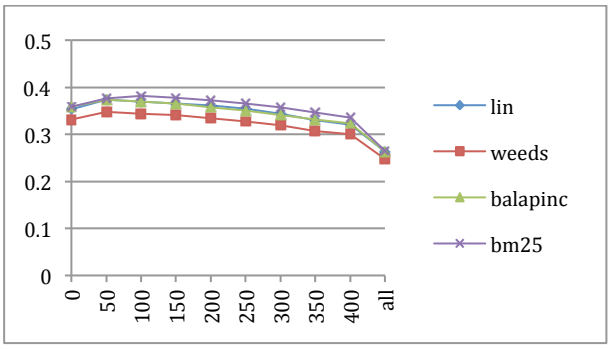


**Figure 3. The effect of *a* on MAP.**

One of the factors that negatively affect the performance is mismatch between the multiword dish names labeled by annotators and the NPs output by the NP chunker. For instance, there are many dish names with prepositions, such as "fish with green curry in banana leaf", which the NP chunker splits into three separate NPs. We are working on a method that aims to identify the correct boundaries of a compound dish name, but this method is outside the scope of this paper.

# 6. Parameter analysis

## 6.1 Number of seeds

In this section we analyse the effect of the number of seeds (*num-seeds*) on performance. The values for *num-seeds* were set to 5, 10, 15 and 20. Ten dish name seed sets were randomly generated for each of *num-seeds* of 5, 10, 15 and 20 from the 100 most frequent dish names as labeled by annotators in the 600 restaurant reviews. We evaluated this parameter with two best performing similarity measures: BM25 and BalAPinc (Figure 4).

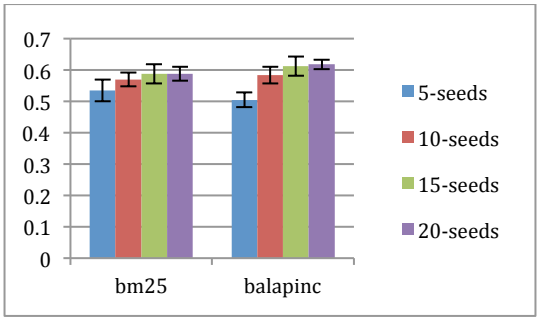


**Figure 4. MAP values of runs with different numbers of seeds**

The use of 10 seeds leads to statistically better results (paired t-test) than the use of 5 seeds with both BM25 ($p<0.03$) and balAPinc ($p<0.01$). The use of 15 seeds is only statistically better than 10 with balAPinc ($p<0.02$), while the use of 20 seeds has a negligible improvement overall. We therefore, conclude that the use of 10 to 15 seeds is sufficient, and that having more seeds does not lead to noticeable improvements.

## 6.2 Discount factors in NPscore

When we combine scores of single nouns to rank noun phrases (Section 4.2), we propose to down-weight their scores the further away they are from the head of the NP. In this section, we compare four discount factors (0.5 discount, log-linear, linear, no discount), as presented in Table 1. Figure 5 shows the comparison of the discount factors applied on 10 testing seed sets with the same tuning parameters as in Table 3.

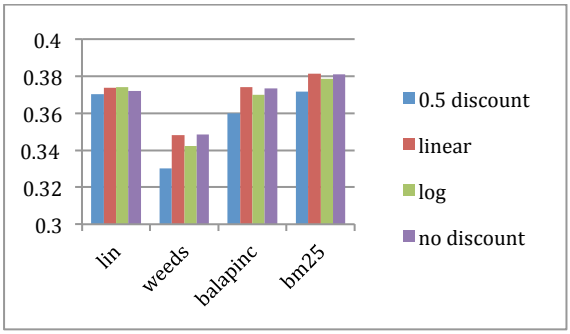


**Figure 5. Average precision of runs with different discount factors.**

There appears to be a small benefit in discounting scores of single nouns based on their distance from the end of the NP when used with some similarity measures. When Lin's similarity function is used, both linear and log discounts are better than no discount (both differences are statistically significant, paired t-test, $p<0.001$). With BalAPinc and BM25 methods, linear discount also works slightly better than no discount (with balAPinc the difference is significant, paired t-test, $p<0.001$), but when Weeds' similarity function is used, applying no discount is better.

# 7. Conclusion

In this paper we presented a method for identifying both single words and multiword units (MWUs) belonging to the same semantic class of entities as a small number of seeds. The method is evaluated on the task of extracting dish names from restaurant reviews. The described method initially computes distributional similarity between each seed, representing a dish name, and each single noun in the corpus, and then produces a list of single nouns, ranked by similarity to all seeds. In parallel, the same method is applied to obtain a list of adjectives ranked by similarity to a set of subjective adjectives. To get and rank MWUs, first, noun phrases (NPs) are obtained from the corpus using an NP chunker, which are then cleaned by removing learned subjective adjectives, and ranked by combining the scores of learned single-noun dish names. We evaluated three distributional similarity measures (Lin's, Weeds' and balAPinc), and propose a new measure, based on the adaptation of an Information Retrieval model, BM25. The proposed BM25-based measure proved to be competitive, and showed statistically significant improvements over the other measures on some of the tasks. Also, the final ranking of MWU dish names was better compared to the other three measures.

In the future we plan to analyse the effect of different types of dependency relations on performance, and see if exclusion of any specific relation types has a positive effect. Another area for future work is improvement of the MWU extraction method. Many MWUs in specialised lexicons such as food/dish names are more complex than the NPs output by NP chunker, therefore a more accurate method of detecting MWU boundaries is needed.

# 8. ACKNOWLEDGEMENTS

I thank We-Create Inc. for providing the corpus of restaurant reviews. I also thank Kaheer Suleman and Mohamad Ahmadi for annotating the reviews.

# 9. REFERENCES

1. Lin, D. Automatic retrieval and clustering of similar words. In: 17th international Conference on Computational Linguistics, pp. 768-774 (1998)
2. Weeds J. and Weir D. A general framework for distributional similarity. In: Conference on Empirical Methods in Natural Language Processing (EMNLP'03), pp. 81-88 (2003)
3. Kotlerman L., Dagan I., Szpektor I., Zhitomirsky-Geffet M. Directional Distributional Similarity for Lexical Expansion. In: ACL-IJCNLP, Singapore, pp. 69-72 (2009)
4. Robertson S.E., Walker S., Jones S., Hancock-Beaulieu M., Gatford M. Okapi at TREC-3. In: Third Text Retrieval Conference, pp.109-126 (1995)
5. Tsai R. T. and Chou C. Extracting Dish Names from Chinese Blog Reviews Using Suffix Arrays and a Multi-Modal CRF Model. In: First International Workshop on Entity-Oriented Search, ACM SIGIR (2011)
6. Rahman A. and Ng V. Inducing Fine-Grained Semantic Classes via Hierarchical and Collective Classification. In: 23rd International Conference on Computational Linguistics (COLING) Beijing, China, August, pp. 931-939 (2010)
7. Riloff, E, and R Jones. Learning dictionaries for information extraction by multi-level bootstrapping." In: 16th National Conference on Artificial Intelligence (1999)
8. Thelen, M. and Riloff E. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In: ACL-02 conference on Empirical methods in natural language processing (EMNLP), USA, 214-221 (2002)
9. Agichtein E. and Gravano L. Snowball: Extracting relations from large plain-text collections. In Proceedings of the 5th ACM Conference on Digital Libraries (2000)
10. Weeds J. and Weir D. Co-occurrence retrieval: a flexible framework for lexical distributional similarity. Computational Linguistics, 31(4), 429-475 (2006)
11. Hearst M. Automatic acquisition of hyponyms from large text corpora. In: the 14th Conference on Computational Linguistics, Nantes, France, 1992.
12. Wang R.C. and Cohen W. Automatic Set Instance Extraction using the Web. In: ACL-IJCNLP, Singapore. 2009.
13. Etzioni O. et al. Unsupervised named-entity extraction from the web: an experimental study. Artificial Intelligence, 165(1), 2005, pp. 91-134.
14. Kozareva Z., Riloff E. and Hovy E. Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. In: ACL-08: HLT, Columbus, USA, 2008.
15. Church K., Gale W., Hanks P., Hindle D. Using statistics in lexical analysis. In: Zernik U., ed. Lexical Acquisition: Using On-line Resources to Build a Lexicon. Englewood Cliffs, NJ, Lawrence Elbraum Associates, pp. 115-164 (1991)
16. Manning C. and Schütze H. Foundations of Statistical Natural Language Processing, MIT Press (1999)
17. Dunning, T. Accurate methods for the statistics of surprise and coincidence. Computational Linguistics, 1993, pp. 61-74.
18. Riloff, E., and Shepherd J. A corpus-based approach for building semantic lexicons. In: Second Conference on Empirical Methods in Natural Language Processing (EMNLP'97), pp. 117-124 (1997)
19. Yarowsky, D. Word sense disambiguation using statistical models of Roget's categories trained on large corpora. In: Fourteenth International Conference on Computational Linguistics (COLING-92), pp. 454-460 (1992)
20. Pantel P., Crestan E., Borkovsky A., Popescu A. and Vyas V. Web-Scale Distributional Similarity and Entity Set Expansion. In: Conference on Empirical Methods in Natural Language Processing, pp. 938-947. Singapore (2009)
21. Kilgarriff A. and Yallop C. What's in a thesaurus? In Proceedings of the Second International Conference on Language Resources and Evaluation, pp. 1371-1379 (2000)
22. de Marneffe M., MacCartney B. and Manning C. Generating Typed Dependency Parses from Phrase Structure Parses. In: Language Resources and Evaluation Conference (2006)
23. Spärck Jones, K., Walker, S., & Robertson, S. E. A probabilistic model of information retrieval: Development and comparative experiments. Information Processing and Management, 36(6), 779–808 (Part 1); 809–840 (Part 2) (2000)