

Helping Editors Choose Better Seed Sets for Entity Set Expansion

ABSTRACT

Gazeteers of named entities are used heavily at commercial search engines such as Google, Yahoo and Bing. Acquiring sets of entities typically consists of combining semi-supervised expansion algorithms with manual cleaning of the resulting sets. In this paper, we study the effects of different seed sets in a state-of-the-art semi-supervised expansion system and show a tremendous variation in expansion performance depending on the choice of seeds. We further show that human editors, in general, provide very bad seed sets, which perform well-below the average random seed set. Finally, we propose various automatic systems for improving editor-generated seed sets, which seek to remove ambiguous and other error-prone seed instances. An extensive experimental analysis shows that expansion quality, measured in R-precision, can be improved on average by a maximum of 46% by removing *the right* seeds from a seed set. Our automatic methods outperform several baselines and on average improve expansion performance by up to 34% over the original seed sets.

Categories and Subject Descriptors

I.2.6 [Artificial Intelligence]: Learning—*concept learning*

General Terms

Algorithms, Experimentation, Measurement.

Keywords

Seed set expansion, information extraction, seed set refinement.

1. INTRODUCTION

Collections of named entities are used in many commercial and research applications, such as question answering [29] and biomedical information extraction [23]. Search engines such as Yahoo, Bing, and Google collect large sets of entities [14, 6] to better interpret queries [27], to improve query suggestions [5] and to understand query intents [11].

Manually creating and maintaining large lists of entities is expensive and laborious. In response, many automatic and semi-automatic techniques have been developed. Among them, semi-supervised techniques are most popular because they allow the users to expand specific target classes without the need for large amounts of training data. Semi-supervised methods, also called set expansion or list expansion range in complexity from simple techniques using lexico-syntactic patterns [10] to more complicated algorithms using distributional techniques [15].

Typical semi-supervised methods such as set expansion, start with a small set of seeds from a specific concept, usually obtained from an editor. Then they use an external source of knowledge such as large corpora of text to expand the set of seeds to a larger set of candidates from the same concept. For example, the seed set {*Oxygen, Mercury, Silicon*} for the concept *Chemical Elements* should return as candidates all the elements in the periodic table. However, even the state of the art systems inevitably produce expansions containing errors and omissions. A particular effort towards cleaning expansion errors is described in [28].

In practice, we observe that the quality of the expansion, for a given system can vary based on the nature of the concept and the seed set. Even within a specific concept, different seed sets can produce widely varying results. This variation in results can be attributed to a variety of factors such as ambiguity of the seeds, noise induced by representation of words within a system, and the *set effect* of seeds, where the given seed set fails to faithfully represent the concept. For example, the seed set {*Helium, Neon, Argon*} is an extremely biased representation for the concept *Chemical Elements*, because it does not completely represent the concept *Chemical Elements*, but tends to represent the more specific concept *Inert Gasses*. In practice, biases might not be as extreme as in the above example, but are still significant enough to introduce errors and omissions in expansions. We also observe that when human editors are asked to give seed sets, the expansion quality is generally .

In this paper we study the impact of different seed sets. From six benchmark lists, we produce five thousand random trial seed sets of size ten each. We show the large variance in their expansion performance and compare with seed sets provided to us by five editors and show that an average user generated seed sets is often below an average random seed set and can thus be improved via automatic techniques. We

proceed by studying in detail, the effect of removals from the editor’s seed sets. We empirically show, by means of exhaustive evaluation that we can improve the quality of seed sets, measured using R-precision[2], by a maximum of 46 percent. We discuss the idea of faithful representations of concepts within the general framework of information theory and provide an unsupervised algorithm from improving the quality of expansions by removing seeds from the original seed sets. The algorithm leverages the intuition that seed sets with un-ambiguous seeds and wider coverage of the concept space tend to represent the concept better. Following [19], we select a subset of seeds that minimizes the semantic overlap between its elements to maximize coverage and minimize ambiguity. We show empirical evidence that we achieve an average improvement in R-precision of 34% using our algorithm.

The remainder of the paper is organized as follows. In the next section we review related work and position our contribution within its landscape. In Section 3, we study the seed sets provided to us by users and exhaustively evaluate all possible removals to empirically show the wide variation in the expansion quality of the seed sets. Section 4 presents our task of seed set refinement within the general framework of information theory and noisy communication and describes an unsupervised algorithm based on [19] to select subsets of editorially generated seed sets. The datasets, our evaluation methodology and our experimental setup is described in Section 5 and in section 6, we present our experimental results and perform an error analysis of our work. Finally we conclude with some discussion and future work in section 7.

2. RELATED WORK

Automatically building large lists of named entities is a common task within information retrieval and natural language processing, supported by a large body of work. Various supervised, unsupervised and semi-supervised techniques have been proposed for this task. Supervised techniques work by tagging occurrences of named entities in text with coarse grained class labels such as *People, Organization and Locations* [12, 9]. Performing finer grained entity tagging with supervised methods require a lot of training data. Unsupervised methods, on the other hand, rely on targeted lexico-syntactic patterns, clustering techniques and word co-occurrence patterns to extract entity sets [18, 8]. However, due to their unsupervised nature, the concepts and entity sets discovered by such methods cannot be targeted to a specific class of interest.

In practice, semi-supervised approaches are commonly used as they allow for targeting a specific class of interest, without the need for more training data. These methods depend on a small set of seeds to create entity lists. They are either based on distributional approaches or use lexico-syntactic patterns to expand a seed set to a larger set of candidates. Some methods such as [22, 21] apply lexico-syntactic patterns to corpora such as web text or query logs, to expand a small set of seeds into a larger set of candidates. Other methods such as [13, 16] use the distributional hypothesis to expand seed sets.

Errors in set expansions are common place and even state of

the art set expansion systems do not generate perfect lists. However, lists with high precision are important in a variety of applications. For example, most search engines use such lists in query interpretation to provide highly relevant results. Editorial annotations are commonly used, sometimes in addition to semi-supervised techniques [28] to clean entity sets generated by set expansion. Using user feedback is a common technique to improve system performance with minimal supervision. For example active learning methods use a group of classifiers to focus the efforts of an editor to test cases which have the maximum impact. Active learning techniques have been successfully used in a variety of natural language tasks[7, 3]. We can imagine an active learning based system which identifies bad seeds in set expansion with help from an editor.

Another approach to generating good seed sets is using completely unsupervised clustering techniques such as CBC [20], and using the committee members generated by CBC as seed sets. Even though, the committee members are unambiguous instances from the concept and can form good seed sets, the concept that is discovered might not be what an editor wanted. Unsupervised methods, by definition are not easy to direct, and hence making it more difficult to target specific concepts.

Seed sets represent concepts. Understand what makes a good seed set, requires understanding how people represent concepts with instances. Prototype theory [25] suggests that people represent a concept using prototypical and unambiguous instances from the concept. However, as shown in section 3.3.1, prototypical examples might actually reduce the quality of expansions. Also, there is a notion of *Basic Level* categories [26], which can be defined as partitions on the concept space that are maximally informative. We try to model this in section 4.3 and show that we can generate seed sets which have high quality expansions.

3. IMPACT OF SEED SETS

3.1 Seed Set Composition

The performance of a set expansion system, measured as the quality of its candidate expansion can vary with the set of seeds provided to it and the type of concept that a set expansion system is trying to expand. These variations in performance can arise from a number of factors such as the size of the seed set, the composition of the seed set and the coherence of the concept being expanded. The performance of set expansion as a function of seed size has been extensively studied in [17], where they show that gains are limited for seed sets of sizes greater than 10. In this section, we study the impact of composition of the seed sets on the expansion performance.

To study the effect of seed set compositionality on expansion performance, we use six lists, which are part of a corpus of 50 lists described in [17]. We sampled 5000 random subsets, each of size 10 from the six lists, totaling 30000 trials. Each of these seed sets were expanded using our set expansion system and then evaluated under the R-precision metric. The performance curves (precision vs. rank) are shown in Figure 3. Table 1 shows there is a wide variation in performance, as much as 41%, confirming that seed

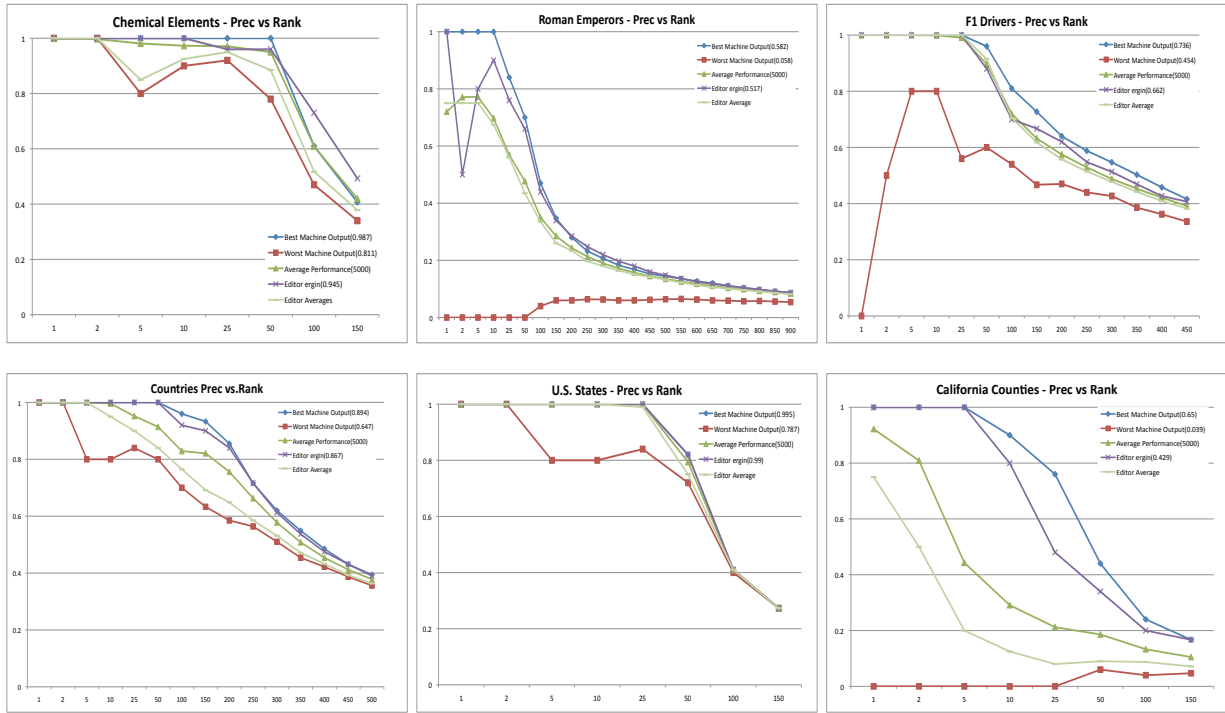


Figure 1: Precision Vs Rank curves for the six benchmark lists

set composition has a significant impact on the quality of expansions.

Table 1: Seed set impacts expansion quality

| List | MAX | MIN | AVG | EDITOR |
|----------------------------|-------|-------|-------|--------|
| <i>Chemical Elements</i> | 0.951 | 0.450 | 0.890 | 0.520 |
| <i>Roman Emperors</i> | 0.374 | 0.0 | 0.283 | 0.283 |
| <i>F1 Drivers</i> | 0.274 | 0.0 | 0.209 | 0.249 |
| <i>Countries</i> | 0.650 | 0.453 | 0.643 | 0.629 |
| <i>U.S. States</i> | 0.952 | 0.538 | 0.913 | 0.876 |
| <i>California Counties</i> | 0.333 | 0.0 | 0.147 | 0.143 |

3.2 Do humans generate good seed sets?

To see how humans compare with the performance of randomly chosen seed sets, we asked four editors to give us ten seed instances from the concepts represented by the six lists. Their seed sets were also expanded under the same conditions (same corpus and set expansion system). Each of the editor scores were averaged and the performance is also shown in Figure 1. We observe that the performance of the average editor is worse than the expansion performance of a randomly chosen set of seeds of the same size. The seed sets given by editors are not good for obtaining high quality expansions.

We also asked an expert editor to provide us with 10 seeds for the same six lists. The performance of expert editor’s seed sets are also shown in Figure 1. An expert editor’s seed sets can perform as good as the best performing random seed set. This observation is also true across the six lists, where the expert editor’s seed sets show consistently higher performance over the average random seed set and the average of non-expert editors. Table 1 shows the r-precision

averaged over the five editors is again below the average random seed set. Table 1 also shows that some lists are harder than others to expand for a fixed size of the seed set, for example, sets such as *F1 Drivers* and *Roman Emperors* are much harder to expand than others, as the difference in R-precision between the maximum and the editor generated seed sets are small.

3.3 Factors in Seed Set Composition

The wide variation in the quality of set expansion due to seed composition can be attributed to three main factors - *prototypicality*, *ambiguity* and *coverage*. While prototypicality and ambiguity are properties of the individual seeds, coverage is a property of the seed set. These factors jointly affect the quality of set expansion. Other factors such as the size and the bias in the corpus used for set expansion and type of the set expansion system used can affect the quality of the set expansion, but are not related directly to the compositionality of seed sets.

3.3.1 Prototypicality

Prototypes are words that are most representative of a class, for example words like *chair* and *table* can be considered prototypical for the class *Furniture*. Most human generated seed sets are composed mostly from prototypical examples, because they are the most natural examples humans use when expressing a concept [24].

In the case of set expansion systems, prototypical examples from a class typically introduce a lot of noise in the expansions. Prototypical examples tend to occur in contexts other not related to their primary sense. For example, a prototypical example for the class of golfers could be *Tiger Woods*,

but in text, the word *Tiger Woods* tends to be used in a lot of other contexts that are not directly related to golfers or golf. This results in a set expansion system considering as candidates that are not just golfers, but other celebrities as well.

All concepts need not have prototypical examples; typically concepts which have prototypes are commonly used classes. Also, prototypes can change if the underlying class changes. For example the class *CEO* can change over time, and thus its prototypes also changes with time. Again, different people tend to have different knowledge bases about a concepts, and prototypes also tend to vary within people. Completely capturing the notion of prototypes is very difficult and requires modeling the underlying class. However, we can approximate the prototypicality of a seed using the probability of drawing a seed from the given concept.

3.3.2 Ambiguity

Seeds which are semantically ambiguous can introduce errors in set expansions. Ambiguous seeds introduce candidates from their other senses into the expansion, reducing the quality of the expanded set. For example, for the class *Chemical Elements*, a seed like *Mercury* can introduce candidates from other classes such as *Roman Gods* and *Planets*. Annotating seed instances with their senses can be impractical in most set expansion systems which depend on a large corpus because that would require annotating all the senses of seed instances within a corpus.

However, we can identify ambiguous seed instances using clustering techniques. Ambiguous seed instances are polysemous and hence similar to other words from the other senses in of the seed. More ambiguous a seed instance is, the lesser is its similarity to the center of any particular sense. Hence they can be identified as the outliers, when a set of seeds are clustered based on their similarities.

3.3.3 Coverage

Another compositional factor which determines the quality of set expansion for a given seed set is the coverage of the seed set. Coverage of a seed set is related to the faithful representation of a concept using a set of instances. A set of instances which can better represent the concept produces expansions which are of better quality than other sets.

Informally, we can define the coverage of a seed set as the size of the concept space included by the seed set. A set of seeds jointly partition a concept space into two parts, one included by the seed set and one excluded by the seed set. Bigger the included partition, more of the intended concept is covered. However, bigger coverage might also lead to the inclusion of a ambiguous seed instances, resulting in an expansion with lower quality. However, coverage can be a useful metric. When considering two sets of seeds which are equal in all other aspects the set with more coverage brings in higher quality results.

4. SYSTEMS

As mentioned in Section 3, on an average, human editors do not produce seed sets which result in good candidate expansions. We also observed that there are three compositional

factors of the seed sets which affect the quality of expansions. To improve the quality of editorial seed sets without any extra effort by the editors, we present three algorithms, each dealing with a single compositional factor.

Removing seed instances from a seed set first requires determining the number of seeds to be removed. The number of seeds removed are parameters to all three algorithms described in this section. In Section 5, we discuss how this parameter was empirically estimated.

4.1 Prototype Removal

The first method deals with removal of prototypes from the seed set. we can statistically approximate the prototypicality score of a seed s as the probability of the word given the concept C from which a word is drawn. This is because prototypical words tend to be more common in text rather than rarer and more specific instances from the same concept. Formally we define prototypicality of a seed s as

$$Prot(s) = \frac{count(s)}{\sum_{s' \in C} count(s')} \quad (1)$$

For each seed in the seed set, we calculate its prototypicality score as defined by Equation 1. Then we sort all the seeds based on their prototypicality score and remove the seeds which are most prototypical of the class, i.e, seeds with high prototypicality score. The number of seeds to be removed is a parameter to this algorithm and is estimated as described in section 6.1.1

4.2 Clustering

Ambiguous seed instances tend to be less similar to any particular sense than non-ambiguous seeds. Thus, as mentioned in section 3.3.2, we can identify them as outliers in a clustering of the seed set. We remove the ambiguous seed instances by first clustering the seed set, and then choosing the tightest cluster as our seed set, and ignoring all other seeds that are not part of the tightest cluster. We use average-link clustering, a hierarchical clustering algorithm that generates a dendrogram over the set of seeds based upon the average similarity between two clusters.

First, we represent each seed in the seed set by representing them using a distributional feature vector, where each feature in the feature vector is composed of contexts that the seed occurs in. We weight the components of the feature vector using point-wise mutual information between the seed and the context. Then, we compute the pair-wise similarity between all the seeds in the seed set as the cosine similarity between the feature vectors.

Then the tightness of each cluster was calculated as the average cosine similarity between all the elements of a cluster. The dendrogram was cut by fixing the number of clusters, k . This value of k was chosen as described in Section 6.1.1.

4.3 Minimum Overlap Criterion (MOC)

The third compositional factor affecting the quality of expansions is coverage of the set given the concept. We can consider the problem of set coverage as one of faithful representation of the concept given a set of seed instances, where

a better represented concept, in terms of its seed instances leads to better expansion quality.

A concept can be defined as a distribution over all words in the vocabulary. Any set of words, or in our case, a seed set, jointly defines a distribution over concepts. A set of seeds S is said to faithfully represent a concept C , if it can maximize the likelihood of the concept given the set of seeds. However, the problem of directly modeling the concept space or the polysemy of a set of words is intractable. To overcome this intractability, we use a non-parametric technique to identify a subset of fixed size that can best represent the concept.

We start with the hypothesis, that for a fixed size, a set of seeds which can faithfully represent a concept C , is one which provides maximum information to the concept. However, the seeds tend to share a lot of semantic information among themselves. For example, the seeds *silicon* and *germanium* for the class *chemical elements*, overlap semantically in the sense that they are both used as semiconductors, thus having a both the seeds in a set biases the semantics of the set towards the class *semi-conductors* and thus failing to faithfully represent the class *chemical elements*. We can avoid this problem by minimizing the semantic overlap between the seeds or selecting a subset of the set of seeds which have minimum semantic overlap between themselves.

To represent the semantics of a seed we use a vector of distributional features, which are all the contexts which appear with the seed. To represent the semantics of the concept, we start by finding the set of features which are common to all the seeds. However, in practice, these vectors are very sparse and the amount of features shared between all the words is very small, introducing the biases in the representation of concept. To overcome this, we represent the concept with the set of features which are shared between a minimum of two seeds in the seed set.

This feature representation allows us to calculate the amount of semantic overlap between any subset of seeds given the concept as joint information overlap between the seeds given the concept. We follow [19] and define the joint information overlap between a set of seeds S and a concept represented as set of features C as

$$I(S; C) = - \sum_{f \in C} p(f) \cdot \log\left(\frac{p(f \wedge S)}{p(S)}\right) \quad (2)$$

where $p(f)$ is the probability of seeing the feature (context) in the corpus, $p(S)$ is the probability of seeing the seed set under consideration, which can be computed as $\sum_{s \in S} p(s)$. $p(f \wedge S)$ is the joint probability of seeing the seed set S and the feature f . Since, our features are contexts surrounding the seeds, they are mutually exclusive. So, we can compute the joint probability as

$$p(f \wedge S) = \sum_{s \in S} p(f, s) \quad (3)$$

Combining equations 2 and 3, we can formally write the

joint information overlap as

$$I(S; C) = \sum_{f \in C} p(f) \cdot \left(\log\left(\sum_{s \in S} p(s)\right) - \log\left(\sum_{s \in S} p(f, s)\right) \right) \quad (4)$$

Finding the set S , which minimizes equation 4 requires evaluating the function over a large discrete state space, which is computationally very expensive. So, we use a greedy algorithm, called MOC, for Minimum Overlap Criterion, that starts with an empty set S and grows the set one seed at a time, at each iteration, choosing the seed which maximizes equation 4 till a fixed number of seeds(k) are in the set. The value of k was determined using a separate development set as described in section 6.1.1

5. DATASETS AND BASELINE

To test the the hypothesis presented in section 4, we selected nine lists of named entities, originally extracted from wikipedia’s *List of pages*. These lists include - *Chemical Elements*, *Roman Emperors*, *F1 Drivers*, *Countries*, *U.S. States*, *CA counties*, *First Ladies of the U.S.*, *American Internet Companies* and *Superheros*. These lists were used as the gold standard for development and testing of the algorithms discussed in 4.

Three of the lists, *First Ladies*, *American Internet Companies* and *Superheros* were designated as the development set, over which the number of seeds to be removed, which is a parameter to all our algorithms, were trained. The remaining sets were used to test the expansion performance of the seed sets generated by the methods discussed in section 4.

Wikipedia served as the source corpus for all the algorithms described in Section 4. All articles were POS-tagged using [4] and shallow parsed (phrase chunked) using a variant of [1]. For each word given to us in the seed sets, we use the processed corpus to extract its left and right contexts, which were then used as features to represent the word for the algorithms described in Section 4. To expand the set of seeds, we follow [17] and use a distributional set expansion algorithm which uses wikipedia as the corpus for extracting other named entities similar to the seed set.

6. EXPERIMENTAL RESULTS

6.1 Experimental Setup

To obtain an upper bound on the expansion performance of the seed set refinement algorithms discussed in Section 4 we performed an exhaustive analysis. First, we created trial seed sets from the original seed sets provided to us by the editors by removing all possible combinations of seeds, of sizes ranging from 1 through 9. This resulted in 1024 trial seed sets for each list in our collection resulting in a total of 9216 trials. Each of these trials were expanded using our distributional set expansion algorithm and their expansions evaluated for their R-precision.

6.1.1 Training the parameters

Two of our algorithms, prototype removals and MOC require the number of seeds, n , to be removed as a parameter. To find the best value we generated trials with various values of removals from 1 through 9 using the development set

Table 2: Extrinsic Overall Results

| System | Elements | Roman Emperors | F1 Drivers | Countries | U.S. States | CA Counties | Average |
|------------------|----------|----------------|------------|-----------|-------------|-------------|--------------|
| MAX | 0.643 | 0.382 | 0.255 | 0.718 | 0.952 | 0.338 | 0.548 |
| USER | 0.481 | 0.160 | 0.163 | 0.514 | 0.857 | 0.073 | 0.374 |
| PROTOTYPE | 0.568 | 0.187 | 0.154 | 0.543 | 0.937 | 0.203 | 0.432 |
| CLUSTER | 0.491 | 0.3 | 0.219 | 0.616 | 0.892 | 0.18 | 0.449 |
| MOC | 0.620 | 0.286 | 0.224 | 0.626 | 0.952 | 0.305 | 0.502 |

Table 3: Intrinsic Results for Each Editor

| USER | Elements | Roman Emperors | F1 Drivers | Countries | U.S. States | CA Counties | Avg. R-Prec |
|------------------|----------|----------------|------------|-----------|-------------|-------------|-------------|
| <i>E1*</i> | 0.646 | 0.24 | 0.172 | 0.689 | 0.944 | 0.245 | 0.489 |
| <i>E2</i> | 0.499 | 0.136 | 0.189 | 0.365 | 0.840 | 0.001 | 0.338 |
| <i>E3</i> | 0.407 | 0.078 | 0.158 | 0.558 | 0.919 | 0.055 | 0.362 |
| <i>E4</i> | 0.457 | 0.171 | 0.146 | 0.446 | 0.774 | 0.023 | 0.336 |
| <i>E5</i> | 0.397 | 0.178 | 0.150 | 0.513 | 0.812 | 0.041 | 0.348 |
| PROTOTYPE | | | | | | | |
| <i>E1*</i> | 0.677 | 0.215 | 0.163 | 0.693 | 0.949 | 0.482 | 0.529 |
| <i>E2</i> | 0.503 | 0.184 | 0.179 | 0.424 | 0.949 | 0.087 | 0.387 |
| <i>E3</i> | 0.522 | 0.185 | 0.144 | 0.680 | 0.927 | 0.157 | 0.435 |
| <i>E4</i> | 0.595 | 0.088 | 0.146 | 0.503 | 0.944 | 0.089 | 0.394 |
| <i>E5</i> | 0.546 | 0.264 | 0.142 | 0.419 | 0.920 | 0.203 | 0.415 |
| CLUSTER | | | | | | | |
| <i>E1*</i> | 0.543 | 0.349 | 0.205 | 0.772 | 0.960 | 0.377 | 0.534 |
| <i>E2</i> | 0.569 | 0.243 | 0.258 | 0.484 | 0.880 | 0.033 | 0.411 |
| <i>E3</i> | 0.569 | 0.276 | 0.205 | 0.541 | 0.880 | 0.148 | 0.436 |
| <i>E4</i> | 0.388 | 0.250 | 0.211 | 0.602 | 0.860 | 0.049 | 0.393 |
| <i>E5</i> | 0.388 | 0.382 | 0.216 | 0.683 | 0.880 | 0.295 | 0.474 |
| MOC | | | | | | | |
| <i>E1*</i> | 0.722 | 0.403 | 0.228 | 0.735 | 0.952 | 0.566 | 0.601 |
| <i>E2</i> | 0.593 | 0.333 | 0.252 | 0.517 | 0.952 | 0.189 | 0.472 |
| <i>E3</i> | 0.565 | 0.278 | 0.217 | 0.739 | 0.952 | 0.264 | 0.502 |
| <i>E4</i> | 0.657 | 0.062 | 0.238 | 0.601 | 0.952 | 0.170 | 0.446 |
| <i>E5</i> | 0.565 | 0.354 | 0.187 | 0.538 | 0.952 | 0.340 | 0.489 |

and expanded them. The expanded candidates were evaluated using the gold standard for the development set and chose n as the value which gave the best average r-precision over all three lists of the development set. For the prototype removal algorithm we determined. For the prototype removal algorithm the number of seeds to be removed was determined to be 3 and for the MOC algorithm the size of the final seed set was determined to be 6, a removal of 4 seeds.

The clustering algorithm requires identifying the number of clusters to determine the cut in the dendrogram. This was determined by first using the development set to generate various number of clusters, k , ranging from 1 through 9. The value of $k = 2$, which gave us the best average r-precision on all three lists of the development set was chosen and used in the clustering algorithm on the test set.

6.2 Overall Analysis

For each list, we considered the trial seed set which gave us the maximum r-precision. This is shown in Table 2 in row **MAX**. Row **USER** shows the performance of the unmodified seeds given by the editors, showing there is a maximum possible improvement in R-precision of 46%. This unmodified seed set is used as a baseline against which all our algorithms are compared. The other rows describe the performance of each system, averaged over all the editors.

Row **PROTOTYPE** shows the performance of the prototype removal algorithm. It shows that simply removing prototypes can significantly improve the editors seed sets by as much as 15%. Prototype removals improve the seed

set of an average editor over all the six benchmark lists tested. Row **CLUSTER** shows the performance of the clustering algorithm, which chooses the tightest cluster, ignoring all other seed sets. Again, like prototype removal, removing ambiguous elements through clustering increases the average r-precision over editors across all six lists, giving an average improvement in R-precision of 19%. It shows that considering and removing ambiguous elements from the seed sets improves the performance of editorially generated seed sets.

Row **MOC** describes the performance of the Minimum overlap criterion method described in 4.3. This method shows the highest performance out of all the three methods described in section 4. The method provides an improvement of 34% over the editors original seed sets.

6.3 Intrinsic Analysis of Prototype Removals and Clustering

The two techniques prototype removal and clustering are simple and intuitive methods for improving the quality of the seed set. Table 3 shows both the methods improve over the average r-precision over the baseline for all five editors. The clustering method improves the seed sets by the worst performing editor, *E5* by upto 36%. The expert editor *E1**'s seed set shows the highest absolute value of R-precision, of 0.534, however for an expert editor there is a small gain in r-precision of only 9%. This is because, expert editors are careful not to give any ambiguous seed instances for their seed set. The gains we obtain in the case of an expert editor is mostly because, what might be intuitively an-ambiguous seeds for humans can occur in text in ambiguous ways.

Table 4: Expansion outputs of USER vs MOC

| USER | MOC |
|-----------------|-------------|
| Los Angeles | Madera |
| San Francisco | Napa |
| San Luis Obispo | Sonoma |
| San Bernardino | Solano |
| Sacramento | Tulare |
| Madera | Alameda |
| Merced | Yolo |
| Modesto | Del Norte |
| San Diego | Los Angeles |
| Bakersfield | Merced |
| Visalia | Yuba |
| Napa | Colusa |
| Tulare | Burlingame |
| Sunnyvale | Plumas |
| Sonoma | Tuolumne |
| Mountain View | Tehama |
| Woodlake | Siskiyou |
| Alameda | Calaveras |
| Santa Rosa | San Benito |
| Arroyo Grande | Mendocino |

The above observation also holds true for the prototype removal technique, which improves over the average r-precision over the baseline for all our five editors, with the r-precision gain in the expert editor’s seed set being only 8%. Again, this can be explained by the fact that expert editors are careful not to provide prototypical examples. All other non expert editors have average r-precision gains between 17.5-19%.

6.4 Intrinsic Analysis of MOC

Table 3 shows the performance of the MOC method on a per-editor basis. MOC improves the average r-precision over the six lists for all five editors. *E1** is the expert editor whose performance for the original seed size of 10 was as good as the best random set described in section 3. It shows that MOC can improve the performance of even an expert editor by as much as 22%. It is also the case that the expert editor’s improved set has the maximum absolute value of 0.601, the highest R-precision achieved among all the editors, showing that when starting with good seed sets, provided by an expert editor we can improve upon their original seed sets to generate candidate expansions of even higher quality.

In terms of maximum gain in R-precision, *E5*, who has the lowest average r-precision of all the editors, has an improvement of close to 50%, showing that bad seed sets provided to us by an editor can be significantly improved by considering a subset which can best represent the concept the editor is trying to expand.

MOC’s high performance compared to more intuitive algorithms like prototype removals and clustering is that some aspects of those algorithms are factored into MOC, which tries to minimize semantic overlap between the seed sets by minimizing the joint information overlap. Seeds which are prototypical tend to overlap semantically with almost all seeds in a seed set, by virtue of them being prototypical examples of a class. This leads MOC to choose seeds which

are not prototypical of the class.

In MOC, the concept an editor is trying to expand is represented as a set of distributional features of all the seeds in a seed set. Ambiguous words, even though have little semantic overlap with other seeds in the seed sets, by virtue of being ambiguous do not share a lot of highly informative distributional features with the concept. This results in MOC tending to chose non-ambiguous seed instances from the seed set. This assumption is verified when looking at the selection process applied to the *U.S. states* list.

{*California, Arizona, Nevada, New York, New Jersey, Washington, Hawaii, Texas, Montana, Indiana*}

This seed set by editor *E3* has ambiguous elements such as *Washington* and *New York* which are both cities and states. As expected, MOC does not chose any of the ambiguous elements, instead selecting the following seeds {*Indiana, New Jersey, California, Arizona, Nevada, Montana*}.

MOC also shows dramatic performance improvement for the list *California Counties* with editor *E5*. Table 4 shows the top 20 expansions when expanded using both the user’s original seed set (*Santa Clara, Contra Costa, Fresno, San Mateo, San Joaquin, Stanislaus, Shasta, Humboldt, Mariposa, Riverside*) and MOC’s subset (*Stanislaus, Mariposa, Contra Costa, San Joaquin, Shasta, San Mateo*). MOC has removed several seeds which are more known as cities than counties. This avoids errors such as *Mountain View* and *Sunnyvale* and *Bakersfield*, introduced by seeds like *Santa Clara* which is also a city. Other errors such as *Arroyo Grande* and *Santa Rosa*, which are county heads are avoided in MOC’s for a similar reason. Counties such as *Fresno* and *Mariposa* are ambiguous, because they are also county heads and MOC avoids them.

Thus MOC, by considering both prototypicality and ambiguity of seed instances, discovers a subset which can best represent the concept, and uses this set to as the seed set in set expansion. Subsets chosen by MOC can improve the quality of set expansion, even in the case of expert editors.

6.5 Error Analysis

All three methods show improvement in average r-precision over the baseline scores for all five editors. Also, the MOC algorithm improves the r-precision scores of each editor for each list, with the exception of the list *Roman Emperors* for editor *E4*. Looking at the seed set, we see too sources of errors. The seed set contains the seed *Romulus*, who is the first emperor of the *Roman Republic*. However, in text, the seed *Romulus* is seen with contexts relating to the founding of rome, rather than contexts usually associated with roman emperors. Also, the seed set contains other well known emperors such as *Nero* and *Augustus*, with a broad semantic space introducing ambiguity in our concept representation.

As MOC tries to chose elements which have minimum semantic overlap with each other, it choses elements which are infrequent in the corpus. Rare elements in seed set results in a loss of recall, and consequently loss in r-precision. This is also true for prototype removal because it directly tries to

pick the least frequent elements in the seed set, causing loss in r-precision. In this case, having prototypical examples in the seed set is useful, when all the seeds are infrequent or a when most of the seeds in the seed set have cover a large semantic space. A way to overcome this problem would be to vary the number of seeds to be removed on a per-list basis.

The clustering algorithm which choses the tightest cluster also improves r-precision for all editors for all lists with the exception of the list *Elements* for two editors *E4* and *E5*. Since, for each list the tightest cluster is chosen, for the editor, the tightest cluster for editor *E4* contained two elements {*Gold, Silver*}, both precious metals. The other seeds in the seed set, such as *Iron, Oxygen, Carbon, Uranium, Mercury, etc ...* were conceptually different and could not form a cohesive cluster. Similar behavior can also be observed in the case of editor *E5*, where the cluster was {*Calcium, Potassium*}, both conceptually different from the other elements in the cluster such as {*Nickel, Carbon, Hydrogen, etc..*}. These errors are caused primarily because, simple clustering cannot completely capture the notion of semantic ambiguity.

7. CONCLUSIONS

In this paper we studied the impact that seed set composition has on the quality of set expansions. We showed that the composition of the seed set can significantly affect the performance of the set expansion, by as much as 41%, using a large set of random trials. We also showed that an average editor does not produce seed sets which result in high quality expansions. In some cases their seed sets can be worse than a randomly chosen seed set from the same concept.

We also asked an expert editor who creates and curates large lists of entities to provide us with seed sets of the same size. Seed sets generated by an expert editor can perform as well as the best seed set randomly chosen from the concept. We identified three important factors in seed set composition - *prototypicality, ambiguity* and *coverage* and showed that by considering these factors when creating seed sets leads to higher quality expansions.

We presented three algorithms, each one tackling a single factor affecting seed set composition. The first algorithm removes prototypes identified by their relative frequency in text and we show that we can improve the quality of expansion by as much as 15%. We presented a second algorithm, which removes ambiguous seeds from a seed set by clustering it and then choosing the tightest cluster. This algorithm improves the quality of the expanded candidates by as much as 19%.

The third algorithm tackles the third factor of seed set composition - *Coverage*. We showed that coverage can be seen as a problem in faithful representation of a concept, and we can find a subset of the seed set which best represents the concept by minimizing the semantic overlap among the seeds in the subset. Semantic overlap, can be seen as a case of minimizing the information overlap between seeds and we provided a simple greedy algorithm for minimizing the joint information overlap between seeds. We discussed how this algorithm balances between the prototypicality and ambiguity of seed instances and can choose subsets which can improve the quality of the seed set by as much as 34%.

8. REFERENCES

- [1] S. Abney and S. P. Abney. Parsing by chunks. In *Principle-Based Parsing*, pages 257–278. Kluwer Academic Publishers, 1991.
- [2] J. A. Aslam and E. Yilmaz. A geometric interpretation and analysis of r-precision. In *CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 664–671, New York, NY, USA, 2005. ACM.
- [3] M. Banko and E. Brill. Scaling to very very large corpora for natural language disambiguation, 2001.
- [4] E. Brill. Transformation based error driven learning and natural language processing : A case study in part of speech tagging. *Computational Linguistics*, 24(4):543–565, 1995.
- [5] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. Li. Context-aware query suggestion by mining click-through and session data. In *Proceedings of KDD-08*, pages 875–883, 2008.
- [6] S. Chaudhuri, V. Ganti, and D. Xin. Exploiting web search to generate synonyms for entities. In *Proceedings of WWW-09*, pages 151–160, 2009.
- [7] I. Dagan and S. P. Engelson. Selective sampling in natural language learning. In *In IJCAI95 Workshop On New Approaches to Learning for Natural Language Processing*, 1995.
- [8] D. Downey, M. Broadhead, and O. Etzioni. Locating complex named entities in web text. In *In Proc. of IJCAI*, 2007.
- [9] R. Florian, R. Florian, A. Ittycheriah, H. Jing, and T. Zhang. Named entity recognition through classifier combination. In *In Proceedings of CoNLL-2003*, pages 168–171, 2003.
- [10] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (Coling 1992)*, pages 539–545, Nantes, France, August 1992.
- [11] J. Hu, G. Wang, F. Lochovsky, J. tao Sun, and Z. Chen. Understanding user’s query intent with Wikipedia. In *Proceedings of WWW-09*, pages 471–480, 2009.
- [12] A. McCallum and W. Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 188–191, Morristown, NJ, USA, 2003. Association for Computational Linguistics.
- [13] M. Paşca. Organizing and searching the world wide web of facts – step two: harnessing the wisdom of the crowds. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 101–110, New York, NY, USA, 2007. ACM Press.
- [14] M. Paşca. Weakly-supervised discovery of named entities using web search queries. In *Proceedings of CIKM-07*, pages 683–690, New York, NY, USA, 2007.
- [15] M. Paşca. Weakly-supervised discovery of named entities using web search queries. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 683–690, New York, NY, USA,

2007. ACM.

- [16] M. Paşca and B. Van Durme. Weakly-supervised acquisition of open-domain classes and class attributes from web documents and query logs. In *Proceedings of ACL-08: HLT*, pages 19–27, Columbus, Ohio, June 2008. Association for Computational Linguistics.
- [17] P. Pantel, E. Crestan, A. Borkovsky, A.-M. Popescu, and V. Vyas. Web scaled distributional similarity applied to entity set extraction. In *EMNLP '09*, Singapore, 2009.
- [18] P. Pantel and D. Lin. Discovering word senses from text. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 613–619, New York, NY, USA, 2002. ACM.
- [19] P. Pantel and V. Vyas. A joint information model for n-best ranking. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 681–688, Manchester, UK, August 2008. Coling 2008 Organizing Committee.
- [20] P. A. Pantel. *Clustering by committee*. PhD thesis, University of Alberta, Edmonton, Alta., Canada, 2003. Adviser-Lin, Dekang.
- [21] E. Riloff and R. Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *PROCEEDINGS OF THE SIXTEENTH NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE (AAAI-99)*, 1999.
- [22] E. Riloff and J. Shepherd. A corpus-based approach for building semantic lexicons. In *In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 117–124, 1997.
- [23] T. C. Rindfleisch, L. Tanabe, and J. N. . Weinstein. Edgar: Extraction of drugs, genes and relations from the biomedical literature.
- [24] E. Rosch. Cognitive representations of semantic categories. *Journal of Experimental Psychology: General*, 104(3):192–233, 1975.
- [25] E. Rosch. Classification of real-world objects: Origins and representation in cognition. pages 212–222, 1977.
- [26] E. Rosch. Principles of categorization. pages 27–48, 1978.
- [27] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and wikipedia. In *Proceedings of WWW-06*, pages 1400–1405, 2006.
- [28] V. Vyas and P. Pantel. Semi-automatic entity set refinement. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 290–298, Boulder, Colorado, June 2009. Association for Computational Linguistics.
- [29] R. C. Wang, N. Schlaefel, W. W. Cohen, and E. Nyberg. Automatic set expansion for list question answering. In *EMNLP*, pages 947–954. ACL, 2008.