"More Like These": Growing Entity Classes from Seeds

Luis Sarmento
FEUP, Universidade do Porto
Rua Dr. Roberto Frias, s/n
Porto, Portugal
las@fe.up.pt

Valentin Jijkoun
ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ
Amsterdam, The Netherlands
jijkoun@science.uva.nl

Maarten de Rijke
ISLA, University of Amsterdam
Kruislaan 403, 1098 SJ
Amsterdam, The Netherlands
mdr@science.uva.nl

Eugenio Oliveira
FEUP, Universidade do Porto
Rua Dr. Roberto Frias, s/n
Porto, Portugal
eco@fe.up.pt

ABSTRACT

We present a corpus-based approach to the class expansion task. For a given set of seed entities we use co-occurrence statistics taken from a text collection to define a membership function that is used to rank candidate entities for inclusion in the set. We describe an evaluation framework that uses data from Wikipedia. The performance of our class extension method improves as the size of the text collection increases.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.1 Content Analysis and Indexing; H.3.3 Information Search and Retrieval; H.3.4 Systems and Software; H.4 [**Information Systems Applications**]: H.4.2 Types of Systems; H.4.m Miscellaneous

General Terms

Algorithms, Measurement, Performance

Keywords

Lexical acquisition, List expansion

1. INTRODUCTION

Lexical resources such as thesauri and ontologies form essential ingredients of many intelligent information access applications. Creating, maintaining, and expanding such resources by manual means is a tedious and expensive task. We are interested in one particular lexical acquisition task: the automatic extension of classes. I.e., given a few seed instances of some (unknown) class of entities, we consult a corpus in order to identify similar entities to add to the class. We are keen on corpus-based methods that can be deployed on arbitrary text collections, regardless of the language. We propose and evaluate one such method that uses a small seed set obtained from the user and "grows" those seeds into sets of similar entities, together forming the extension of some concept. An example is provided by the seed set {*Raphael, Michelangelo, Leonardo da Vinci*}, which might be expanded so as to include, e.g., *El Greco*, *Sandro Botticelli*, *Jan van Eyck*, depending on our background corpus. We also describe an evaluation framework for class expansion methods that uses data from Wikipedia.[1] Evaluation of our class expansion method on English and Portuguese Wikipedias indicates that its performance improves as the frequency of candidate class members increases.

CIKM'07, November 6–8, 2007, Lisboa, Portugal.

2. RELATED WORK

Roark and Charniak [4] describe a corpus-based method for expanding a nominal category from a small set of example "seed" words. The method selects new category members by considering co-occurrence with the seed words. The co-occurrence statistics is based on noun conjunctions, lists and appositives. The focus is on common nouns ("*car*", "*pickup trucks*"); no results are reported for named entities. A similar bootstrapping approach due to Thelen and Riloff [5] relies on a large body of extraction patterns that capture information about the behaviour of a word. E.g., patterns such as "*X was arrested*" or "*murdered X*" are likely to extract candidates of the category *People*; a word is considered a good candidate for inclusion in a category if it is extracted by patterns that also extract known category members. Ghahramani and Heller [3] propose a probabilistic Bayesian framework for expanding a class from seed entities. The method estimates the probability that a candidate belongs to a (hidden) class, based on the available information. The authors compare their class expansion algorithm to Google Sets[2] and show a significant improvement. Finally, the task we are addressing is similar to the List Completion task that is to be evaluated at INEX (Initiative for the Evaluation of XML Retrieval) [2, 1].

3. MEMBERSHIP FUNCTIONS

The task we are address is the expansion of entity classes: given a set S of entities (seeds) that belong to some class

[1] `http://www.wikipedia.org`
[2] `http://labs.google.com/sets`

C, and a set E of candidate entities, we wish to determine which of the entities in E belong to C. In other words, we want to "grow" a class C from a few seed examples, choosing elements from E. Consider, for example, the seed S_1={*Michelangelo*, *Leonardo da Vinci*}. This seed may, in principle, be used to describe any of the classes C_1: {"Renaissance artists"}, C_2: {"Italian artists"} or C_3: {"Italian people"}. Binary classification cannot deal with such descriptive vagueness in a straightforward way. Instead, we approach the class expansion problem by defining a *class membership function*[3] $\mu(C, e)$ that computes the degree of membership of entity e to class C (a value between 0 and 1). Since in the class expansion task the actual C is not known, we approximate $\mu(C, e)$ by $\mu(S, e)$: the latter function defines the degree to which e belongs to the same entity class(es) as *all* elements of S. For the example above, we could define $\mu(C_1, \textit{Jan van Eyck}) = 1$ and $\mu(C_2, \textit{Jan van Eyck}) = 0$, whereas $\mu(S_1, \textit{Jan van Eyck})$ would be defined as a value between 0 and 1, probably, closer to 1. At the same time, we would like to have $\mu(S_1, \textit{Linux}) = 0$, as *Linux* shares few (if any) semantic properties with the elements of S_1. If we can compute the value of the membership function $\mu(S, e)$ for a given seed set S and any candidate entity e, we can define the *expansion* of S as the set of n elements of E with the highest values of $\mu(S, e)$. Thus, for given S, and E, we reduce the class expansion problem to the task of computing $\mu(S, e)$ for all $e \in E$.

We consider the membership function as a kind of similarity function between a set of seed entities (S) and a single entity (e), and compute the values of the membership function using the vector space model (VSM). In the VSM, each element (in our case, a seed set of entities or a candidate entity) w_j is represented by a vector of numerical features, $\overline{w_j}$. Given such a representation we can compare elements using standard distance measures (e.g., cosine similarity). The choice of features defines the information that is captured and transferred to the vector space. We are mainly targeting *type similarity*. Based on the fact that humans easily group and list type similar objects, we assume that many explicit enumerations of entities that we find in text are lists of objects of similar classes. Therefore, if we can identify such enumerations in text, we may be able to gather information about class similarity. Our assumption (also used in, e.g., [4, 5]) is that if two elements consistently co-occur in lists, they are likely to be of a similar semantic class.

Identification of lists in text, however, may not be trivial. We propose a simple approximation that tries to identify *pairs* of elements that belong to textual lists. We assume that lists are composed by sequences of pairs of coordinated elements, which are either connected by explicit coordinational elements ("*and*", "*or*" ...) or by commas. This intuition has been corroborated by studies that focus specifically on using information about coordinated words (see, e.g., [6]). In order to identify entity pairs that belong to lists, we look for structures like "... ne_a, ne_b *and* ne_c ..." where ne_a, ne_b, etc. are named entities. E.g., "*I've lived in NY, Paris, and Amsterdam.*" Another possibility would be "... ne_a, ne_b *or* ne_c ..." as in "*Experience with Oracle, PostgreSQL or MySQL is required*"). When instances of such patterns are found in a corpus, we can easily conclude that the pairs ("ne_a","ne_b") and ("ne_b","ne_c") co-occur in coordination.

[3]We borrow the terminology from Fuzzy Set Theory.

Note that, for simplicity, we will not make any conclusions about "ne_a" and "ne_c". Having extracted such co-occurence information for all entities in the corpus, we can represent our entities and entity sets as vectors encoding the frequency of co-occurence. Specifically, if $ne_1, \ldots, ne_N$ are all (named) entities in the corpus that occur in coordination constructions, then the j-th component of the vector $\overline{ne_i}$ is defined as the number of times ne_i and ne_j co-occur in coordination. Similiarly, for an entity set S_k, $\overline{S_k}(j)$ is defined as the number of times ne_j co-occurs with any element of S_k in coordination. However, pairs of elements connected by comma are subject to a lot of noise because surface text structures of the form "... X, Y, ..." may occur often without implying any class similarity between X and Y, but introduce apposition, or clarification, instead. Therefore, an alternative is to take only information about pairs connected by explicit coordinations such as "... ne_a *and* ne_b ..." or "... ne_a *or* ne_b ...". The feature vectors generated with this restriction may be less noisy, though we are likely to hurt recall. We will refer to vector spaces built using only explicitly coordinated pairs by $\mathcal{VS}^X$, and to those built from explicit coordinations *and* commas as $\mathcal{VS}'$.[4]

With a given feature representation and vector space, the membership function between an entity $\overline{e}$ and an entity set $\overline{S}$ can be defined based on any standard distance measure for vector spaces. In our experiments we use the *cosine similarity measure*, so that $\mu(S, e) = \cos(\overline{S}, \overline{e})$. Depending on the vector space used to obtain the vector representation of the entities and sets, we can have different definitions of the membership function. We will differentiate between μ', which is calculated using $\mathcal{VS}'$, and μ^X which is calculated using $\mathcal{VS}^X$. We will use μ when the difference is immaterial.

4. EVALUATION USING WIKIPEDIA

Our motivation for using Wikipedia to develop the evaluation framework is two-fold. First, Wikipedia contains several explicit human-generated lists that can serve as gold standards for class expansion algorithms. Such lists correspond both to fairly common concepts (e.g., "*List of English novelists*"), as well as rather exotic and very specific lists (e.g., "*List of fractals by Hausdorff dimension*").[5] We can thus obtain evaluation data from different domains and different levels of specificity. Second, Wikipedia has explicit information regarding the *delimitation* of named entities in text. Many articles in Wikipedia are about named entities, and articles explicitly link to each other. Whenever in the text of an article A_1 we find a (hyper)link to another Wikipedia article A_2 such that the anchor text of the link starts with a capital letter, we may assume that the anchor text is a mention of an entity, the one addressed in A_2. Using this heuristic, we automatically identify correctly delimited mentions of named entities in Wikipedia articles. We thus circumvent the problem of *identifying* named entities in text, and focus on evaluating membership functions and class expansion algorithms.

In the definition of the class expansion task (Section 3), the input of a system contains the seed set S and the set of candidates E from which we will chose the ones that belong

[4]Importantly, when applying our method, we will always be able to guarantee that "X" and "Y" are in fact named entities (see Section 4).

[5]See `http://en.wikipedia.org/wiki/`*name of the list*.

to the (unknown) class C, which is implicitly defined by the seed examples. Since the Wikipedia lists that we will use to create our evaluation sets are obviously not guaranteed to be complete and exhaustive (we can not be sure that all valid elements are included), instead of directly evaluating the class expansion task, we will evaluate the quality of rankings of E that our membership functions produce. Specifically, for a gold standard class C, which corresponds to a Wikipedia list, we will start by picking a subset that contains positive examples $\mathcal{P}$ ($\mathcal{P} \subset C$). Then, we will construct the set $\mathcal{N}$ that contains negative examples, that should include entities that are somehow *related* to elements of C, but do not belong to C. In general, $|\mathcal{N}| \gg |\mathcal{P}|$. For a given C and sets of *positives* $\mathcal{P}$ and *negatives* $\mathcal{N}$, a *test case* consists of a seed set taken from $\mathcal{P}$, $S \subset \mathcal{P}$. To evaluate the quality of a membership function μ on this test case, we construct the set of candidates $E = \mathcal{P} \cup \mathcal{N} \setminus S$, rank the elements E by $\mu(S, \cdot)$ and assess the quality of the resulting ranking R using *average precision*, a measure commonly used to evaluate the quality of a ranking:

$$AP(S,R) = \frac{\sum_{r=1}^{|E|} P_{at}(r,R) \cdot \mathcal{I}(R(r) \in \mathcal{P} \setminus S)}{|\mathcal{P} \setminus S|}. \quad (1)$$

Here, $P_{at}(r)$ is the value of precision at rank r (i.e., the number of positive examples among $R(1), \ldots, R(r)$, divided by r), and $\mathcal{I}(x \in X)$ is the indicator function of set membership (1 if $x \in X$, and 0 if $x \notin X$). Average precision favors rankings that produce relevant items (in our case, positive examples) earlier. A *test set* is a collection of all test cases for a given C, $\mathcal{P}$ and $\mathcal{N}$. The overall quality of a membership function μ on a test set can be assessed using the *Mean Average Precision* of the rankings produced for all test cases:

$$MAP = \frac{\sum_{i=1}^{m} AP(S_i, R_i)}{m}, \quad (2)$$

where m is the number of test cases and R_i is a ranking produced for the test case i with seed S_i.

The construction of the actual test sets is done in two stages, and is based on XML encoded dumps of Wikipedia.[6] We produced test sets using the English XML dump (1.6M files, 2.9Gb) and the Portuguese XML dump (0.21M files, 227MB). First, we detected lists in Wikipedia pages. Such pages can be easily identified by an expressive title: "List of ..." for English and "Lista de ..." for Portuguese. We only considered the subset of those pages which present information using explicit *HTML list structures*, because they are easy to process (i.e., tables are ignored). Then, for each element of the extracted lists we obtain (i) the corresponding frequency in Wikipedia (i.e., number of times that it occurs as a link, as explained in previously), and (ii) the URL of the Wikipedia article that addresses that entity. For the elements that do not posses a corresponding article in Wikipedia (i.e., point to a page were the user is invited to start an article about that topic) we still extracted the element but we kept information regarding the absence of the article. Entirely numeric elements are ignored, to avoid long lists of dates or other numerical values (e.g., telephone codes) which are useless for evaluation purposes. We were able to extract 17,594 lists for English (referred to as $\mathcal{L}_{en}$) and 1,390 lists for Portuguese ($\mathcal{L}_{pt}$). Lists from $\mathcal{L}_{en}$ contained on average 92.4 elements, and 58.4 elements on average were linked to a existing Wikipedia article. For $\mathcal{L}_{pt}$ the numbers are 90.3 and 43.4, respectively. One major difference between lists in $\mathcal{L}_{en}$ and $\mathcal{L}_{pt}$ is the average frequency of their elements throughout the corresponding Wikipedias. Elements in $\mathcal{L}_{en}$ have an average frequency of 286.2, whereas for $\mathcal{L}_{pt}$ it is only 32.5.

We generated the sets of positives, $\mathcal{P}$, and the corresponding sets of negatives, $\mathcal{N}$, from $\mathcal{L}_{en}$ and $\mathcal{L}_{pt}$ as follows. We first selected only those lists which contained a minimum number of elements, $\min_a$, with a dedicated Wikipedia article. This filtering was done to guarantee that the topic addressed by the list is reasonably well covered in Wikipedia. We set $\min_a = 10$ for English and Portuguese and we thus obtained two more restrictive sets of list, $\mathcal{L}_{en}^{\min_a}$ and $\mathcal{L}_{pt}^{\min_a}$. Then for each list $\lambda(i)$ in $\mathcal{L}_{en}^{\min_a}$ and in $\mathcal{L}_{pt}^{\min_a}$ we chose all items that have both a dedicated article and whose frequency in Wikipedia is higher than a given threshold $f_{\min}$. These will constitute $\mathcal{P}_{cand}(i)$, the candidates for set $\mathcal{P}(i)$. Next, for each element in $\mathcal{P}_{cand}(i)$ we extract all entities from the corresponding Wikipedia articles. Such entities are added to set $\mathcal{N}_{cand}(i)$, except those that belong to list $\lambda(i)$. The set $\mathcal{N}_{cand}(i)$ will thus consist of all sorts of entities "related" to elements from $\mathcal{P}_{cand}(i)$ but which are known not to belong to the initial list $\lambda(i)$. Since sets $\mathcal{P}_{cand}(i)$, $\mathcal{N}_{cand}(i)$ can be extremely large the final $\mathcal{P}(i)$ and $\mathcal{N}(i)$ test sets are obtained by truncating the candidate sets. Thus, only the top $n_{\mathcal{P}}$ most frequent elements from $\mathcal{P}_{cand}(i)$ are chosen (if there are less than $n_{\mathcal{P}}$, all are chosen). These will become set $\mathcal{P}(i)$. Again, we are trying to ensure that μ is tested on sufficiently frequent items. From $\mathcal{N}_{cand}(i)$, we also chose the top $n_{\mathcal{N}}$ most frequent elements with $n_{\mathcal{N}}$ being set to twice the number of elements in $\mathcal{P}(i)$. These elements will become set $\mathcal{N}(i)$ Finally, we exclude all $\mathcal{P}(i)$ and $\mathcal{N}(i)$ sets for which the number of elements in $\mathcal{P}(i)$ is less that 5. We set $f_{\min} = 100$ both for English and Portuguese. Also, in both cases we set $n_{\mathcal{P}} = 20$. For practical reasons, this number can not be higher because the test procedure will involve combinations of elements from $\mathcal{P}(i)$. We were able to generate 3,219 test sets for English and 75 for Portuguese. In both cases, only about one third of the tests do actually reach the $n_{\mathcal{P}}$ limit. Again, the major difference is the average frequency of the elements contained in the $\mathcal{P}(i)$ sets, measured over the corresponding Wikipedia. For English, that figure is 1,758.3 while for Portuguese it is 623.6, which is still high given the relative sizes of the two Wikipedias.

We collected pairs of coordinated named entities for English and for Portuguese using the previously described XML dumps. As explained before, an important reason for having chosen Wikipedia as the source corpus for grounding our membership function μ is that we can thus avoid the complex problem of identifying/delimiting named entities in text. By using a simple heuristic based on links found in Wikipedia, we can easily identify named entities in the articles. We should emphasize that the only textual information used for extracting the pairs of coordinated named entities was the text contained in paragraphs inside articles (8.8M paragraphs for English and 0.76M for Portuguese). Category and language links, information boxes and explicit list information (both lists and tables) were ignored. Paragraphs in Wikipedia articles were scanned for structures of the form "(ne_a) (*coordination connector*) (ne_b)". We extracted features defining four vector spaces: $\mathcal{VS}_{EN}^{X}$

[6] Available from the University of Amsterdam: `http://ilps.science.uva.nl/WikiXML`.

and $\mathcal{VS}^X_{PT}$, using only explicit coordination, and $\mathcal{VS}'_{EN}$ and $\mathcal{VS}'_{PT}$, using both explicit coordinations and comma (Table 1). For English, we used the following explicit coordination connectors "*and the*", "*and a*", "*and*", "*or the*", "*or a*", "*or*". For Portuguese we used "*e o*", "*e um*","*e a*", "*e uma*", "*e do*", "*e da*", "*e*", "*ou o*", "*ou um*", "*ou a*", "*ou uma*", "*ou*".

	NE Pairs	Distinct NE Pairs	Dim($\mathcal{VS}$)
$\mathcal{VS}'_{EN}$	2,172,790	1,255,204	819,379
$\mathcal{VS}^X_{EN}$	1,755,603	516,415	500,980
$\mathcal{VS}'_{PT}$	154,836	119,174	85,494
$\mathcal{VS}^X_{PT}$	44,919	36,751	46,601

Table 1: NE's extracted and Vector Spaces

We conducted evaluation of both μ^X and μ' for English and Portuguese over all pairs of sets $\mathcal{P}(i)$ and $\mathcal{N}(i)$ obtained for each language. Each $\mathcal{P}(i)$ set had up to a maximum of 20 elements and each $\mathcal{N}(i)$ set had exactly twice as many elements as the corresponding $\mathcal{P}(i)$. Due to the combinatorial nature of the evaluation procedure we restricted the size of the seed sets to only two elements. For each pair of $\mathcal{P}(i)$ and $\mathcal{N}(i)$, we generated all possible seed combinations of 2 elements from $\mathcal{P}(i)$. Then, for each seed combination, $\mathcal{S}_k$ we started by obtaining $\overline{\mathcal{S}_k}$, the vector representation of $\mathcal{S}_k$ in the corresponding vector space ($\mathcal{VS}'$ or $\mathcal{VS}^X$). Next, we used μ' and μ^X to compute the degree membership of each of the candidate element e ($e \in \mathcal{P}(i) \bigcup \mathcal{N}(i) \setminus \mathcal{S}_k$) using its representation in the corresponding vector space ($\mathcal{VS}'$ or $\mathcal{VS}^X$). We proceeded by ranking all candidate elements according to the previously computed value of membership and computed the Average Precision (AP) of that ranking. Finally, Mean Average Precision scores were computed using all values of AP obtained for each seed combination.

5. RESULTS

The first two rows of Table 2 contain the average MAP values for μ' and μ^X taken over the complete test sets (3219 for English and 75 for Portuguese). For both languages the average performance obtained by μ' is considerably higher than the performance obtained by μ^X (+0.135 for English and + 0.116 for Portuguese). According to the one-tail sign test the improvement is significant in both cases ($p < 0.0001$ in both cases). For Portuguese the results for both μ' and μ^X are considerably better than for English. One explanation is that the threshold used for selecting the elements included in the set of Positives, $\mathcal{P}(i)$, was the same for English and Portuguese ($f_{min} = 100$) and, thus, relatively higher for Portuguese taken the relative sizes of both Wikipedias. Therefore, the resulting test sets for Portuguese contain relatively more frequent elements, benefiting corpora-based methods such as ours.

In order to further assess the impact of frequency values in the performance of μ, we developed additional test sets for Portuguese by choosing *the least frequent elements* from $P_{cand}(i)$ (keeping $f_{min} = 100$). The corresponding sets of negatives, $\mathcal{N}(i)$ were kept the same. There are only 28 cases in which the new tests are actually different (i.e., $\#\mathcal{P}_{Cand}(i) > 20$ elements). We will denote the 28 new test sets as $\mathcal{P}^-_{28}$. We compared these with the corresponding 28 sets used in the previous experience, which we will denote as $\mathcal{P}_{28}$. The value of $\bar{f}_{avg}$ (average frequency of elements, averaged over all test sets) for $\mathcal{P}^-_{28}$ dropped to 189.4, which clearly indicates that we are now dealing with sets of Positives containing much less frequent elements than in the previous experiment. On the other hand, the corresponding $\bar{f}_{avg}$ value for $\mathcal{P}_{28}$ increased to 982.2. We repeated the evaluation procedure on the newly created test sets, $\mathcal{P}^-_{28}$ and $\mathcal{P}_{28}$. The results are shown in two bottom rows of Table 2. The performance of both μ', μ^X decreases for the test sets ($\mathcal{P}^-_{28}$, $\mathcal{N}$). The decreases occur consistently over the 28 test sets (test sign: $p < 0.01$ for μ' and $p < 0.0001$ for μ^X). The drop for μ^X is more significant, but was expected since the associated vector space $\mathcal{VS}^X_{PT}$ is much smaller than $\mathcal{VS}'_{PT}$ and, thus, the probability of not finding vector representation for some elements in the test set has increased. Moreover, the performance of μ' and μ^X is the highest on the test set ($\mathcal{P}_{28}$, $\mathcal{N}$), which has the highest average element frequency. This indicates that the performance of the membership functions improves as the frequency of the elements to which they are applied increases.

	#tests	$\bar{f}_{avg}$	μ'	μ^X
EN(all)	3219	1758.3	0.424	0.289
PT(all)	75	623.6	0.542	0.426
PT($\mathcal{P}_{28}$, $\mathcal{N}_{28}$)	28	982.2	0.547	0.493
PT($\mathcal{P}^-_{28}$, $\mathcal{N}_{28}$)	28	189.4	0.431	0.229

Table 2: Average values of MAP over all test sets

6. CONCLUSIONS

We presented a corpus-based method for the class expansion task based on a class membership function, estimated from statistics of co-occurrence of named entities in coordination constructions. We also presented an evaluation framework based on entity lists automatically extracted from Wikipedia. We showed that the performance of our method improves as the frequencies of the candidate entities in the text corpus increase (which are related to the corpus size). In future work, we will experiment other association measures for building the vector spaces (e.g. log-likelihood ratio, mutual information). We also plan to compare the performance of our method with [3] and with Google Sets.

7. ACKNOWLEDGMENTS

This work was partially supported by grant SFRH/BD/23590/2005 from FCT (Portugal), co-financed by POSI, and by the Netherlands Organization for Scientific Research (NWO) under project numbers 220-80-001 and 612.000.106.

8. REFERENCES

[1] S. Fissaha Adafre, M. de Rijke, and E. Tjong Kim Sang. Entity retrieval. In *RANLP 2007*, 2007.

[2] N. Fuhr and M. Lalmas. Advances in XML Retrieval: The INEX Inititative. In *IWRIDL 2006*, 2006.

[3] Z. Ghahramani and K. A. Heller. Bayesian sets. In *Advances in Neural Information Processing Systems 18 (NIPS)*, 2005.

[4] B. Roark and E. Charniak. Noun-phrase co-occurrence statistics for semiautomatic semantic lexicon construction. In *COLING-ACL*, pages 1110–1116, 1998.

[5] M. Thelen and E. Riloff. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *EMNLP '02*, pages 214–221, 2002.

[6] D. Widdows and B. Dorow. A Graph Model for Unsupervised Lexical Acquisition. In *COLING 2002*, pages 1093–1099, 2002.