Large-scale Computation of Distributional Similarities for Queries

Enrique Alfonseca
Google Research
Zurich, Switzerland
ealfonseca@google.com

Keith Hall
Google Research
Zurich, Switzerland
kbhall@google.com

Silvana Hartmann
University of Stuttgart
Stuttgart, Germany
silvana.hartmann@ims.uni-stuttgart.de

Abstract

We present a large-scale, data-driven approach to computing distributional similarity scores for queries. We contrast this to recent web-based techniques which either require the off-line computation of complete phrase vectors, or an expensive on-line interaction with a search engine interface. Independent of the computational advantages of our approach, we show empirically that our technique is more effective at ranking query alternatives that the computationally more expensive technique of using the results from a web search engine.

1 Introduction

Measuring the semantic similarity between queries or, more generally, between pairs of very short texts, is increasingly receiving attention due to its many applications. An accurate metric of query similarities is useful for *query expansion*, to improve recall in Information Retrieval systems; for *query suggestion*, to propose to the user related queries that might help reach the desired information more quickly; and for *sponsored search*, where advertisers bid for keywords that may be different but semantically equivalent to user queries.

In this paper, we study the problem of measuring similarity between queries using corpus-based unsupervised methods. Given a query q, we would like to rank all other queries according to their similarity to q. The proposed approach compares favorably to a state-of-the-art unsupervised system.

2 Related work

Distributional similarity methods model the similarity or relatedness of words using a metric defined over the set of contexts in which the words appear (Firth, 1957). One of the most common representations for contexts is the vector space model (Salton et al., 1975). This is the basic idea of approaches such as (Grefenstette, 1992; Bordag, 2008; Lin, 1998; Riloff and Shepherd, 1997), with some variations; e.g., whether syntactic information is used explicitly, or which weight function is applied. Most of the existing work has focused on similarity between single words or syntactically-correct multiword expressions. In this work, we adapt these techniques to calculate similarity metrics between pairs of complete queries, which may or may not be syntactically correct.

Other approaches for query similarity use statistical translation models (Riezler et al., 2008), analysing search engine logs (Jones et al., 2006), looking for different anchor texts pointing to the same pages (Kraft and Zien, 2004), or replacing query words with other words that have the highest pointwise mutual information (Terra and Clarke, 2004).

Sahami and Helman (Sahami and Heilman, 2006) define a web kernel function for semantic similarity based on the snippets of the search results returned by the queries. The algorithm used is the following: (a) Issue a query x to a search engine and collect the set of n snippets returned by the search engine; (b) Compute the tf·idf vector v_i for each document snippet d_i; (c) Truncate each vector to include its m

Proceedings of NAACL HLT 2009: Short Papers, pages 29–32,
Boulder, Colorado, June 2009.

highest weighted terms; (d) Construct the centroid of the L_2-normalized vectors v_i; (e) Calculate the similarity of two queries as the dot product of their L_2-normalized vectors, i.e. as the cosine of both vectors.

This work was followed up by Yih and Meek (Yih and Meek, 2007), who combine the web kernel with other simple metrics of similarity between word vectors (Dice Coefficient, Jaccard Coefficient, Overlap, Cosine, KL Divergence) in a machine learning system to provide a ranking of similar queries.

3 Proposed method

Using a search engine to collect snippets (Sahami and Heilman, 2006; Yih and Meek, 2007; Yih and Meek, 2008) takes advantage of all the optimizations performed by the retrieval engine (spelling correction, relevance scores, etc.), but it has several disadvantages: first, it is not repeatable, as the code underlying search engines is in a constant state of flux; secondly, it is usually very expensive to issue a large number of search requests; sometimes the APIs provided limit the number of requests. In this section, we describe a method which overcomes these drawbacks. The distributional methods we propose for calculating similarities between words and multiword expressions profit from the use of a large Web-based corpus.

The contextual vectors for a query can be collected by identifying the contexts in which the query appears. Queries such as *[buy a book]* and *[buy some books]* are supposed to appear close to similar context words in a bag-of-words model, and they should have a high similarity. However, there are two reasons why this would yield poor results:

First, as the length of the queries grows, the probability of finding exact queries in the corpus shrinks quickly. As an example, when issuing the queries *[Lindsay Lohan pets]* and *[Britney Spears pets]* to Google enclosed in double quotes, we obtain only 6 and 760 results, respectively. These are too few occurrences in order to collect meaningful statistics about the contexts of the queries.

Secondly, many user queries are simply a concatenation of keywords with weak or no underlying syntax. Therefore, even if they are popular queries, they may not appear as such in well-formed text found in web documents. For example, queries like *[hollywood dvd cheap]*, enclosed in double quotes, retrieve less than 10 results. Longer queries, such as *[hotel cheap new york fares]*, are still meaningful, but do not appear frequently in web documents.

In order to use of distributional similarities in the query setting, we propose the following method. Given a query of interest $p = [w_1, w_2, ..., w_n]$:

1. For each word w_i collect all words that appear close to w_i in the web corpus (i.e., a bag-fo-words models). Empirically we have chosen all the words whose distance to w_i is less or equal to 3. This gives us a vector of context words and frequencies for each of the words in the query, $\vec{v_i} = (f_{i1}, f_{i2}, ..., f_{i|V|})$, where $|V|$ is the size of the corpus vocabulary.
2. Represent the query p with a vector of words, and the weight associated to each word is the geometric mean of the frequencies for the word in the original vectors:

$$\vec{qv} = \left(\left(\prod_{i=1}^{|n|} f_{i1} \right)^{\frac{1}{n}}, \left(\prod_{i=1}^{|n|} f_{i2} \right)^{\frac{1}{n}}, ..., \left(\prod_{i=1}^{|n|} f_{i|V|} \right)^{\frac{1}{n}} \right)$$

3. Apply the χ^2 test as a weighting function test to measure whether the query and the contextual feature are conditionally independent.
4. Given two queries, use the cosine between their vectors to calculate their similarity.

The motivations for this approach are: the geometric mean is a way to approximate a boolean AND operation between the vectors, while at the same time keeping track of the magnitude of the frequencies. Therefore, if two queries only differ on a very general word, e.g. *[books]* and either *[buy books]* or *[some books]*, the vector associated to the general words (*buy* or *some* in the example) will have non-zero values for most of the contextual features, because they are not topically constrained; and the vectors for the queries will have similar sets of features with non-zero values. Equally relevant, terms that are closely related will appear in the proximity of a similar set of words and will have similar vectors. For example, if the two queries are *Sir Arthur Conan Doyle books* and *Sir Arthur Conan Doyle novels*, given that the vectors for *books* and *novels* are expected to have similar features, these two queries

Contextual word	acid	fast	bacteria	Query
acidogenicity	11	6	4	6.41506
auramin	2	5	2	2.71441
bacillae	3	10	4	4.93242
carbolfuchsin	1	28	2	8.24257
dehydrogena	5	3	3	3.55689
diphtheroid	5	9	92	16.05709
fuchsine	42	3	4	7.95811
glycosilation	3	2	3	2.62074

Table 1: Example of context words for the query *[acid fast bacteria]*.

	0	1	2	3	4
0	280	95	14	1	0
1	108	86	65	4	0
2	11	47	83	16	0
3	1	2	17	45	2
4	0	0	1	1	2

Table 2: Confusion matrix for the pairs in the goldstandard. Rows represent first rater scores, and columns second rater scores.

will receive a high similarity score.

On the other hand, this combination also helps in reducing word ambiguity. Consider the query *bank account*; the bag-of-words vector for *bank* will contain words related to the various senses of the word, but when combining it to *account* only the terms that belong to the financial domain and are shared between the two vectors will be included in the final query vector.

Finally, we note that the geometric mean provides a clean way to encode the pair-wise similarities of the individual words of the phrase. One can interpret the cosine similarity metric as the magnitude of the vector constructed by the scalar product of the individual vectors. Our approach scales this up by taking the scalar product of the vectors for all words in the phrase and then scaling them by the number of words (i.e., the geometric mean). Instead of computing the magnitude of this vector, we use it to compute similarities for the entire phrase.

As an example of the proposed procedure, Table 1 shows a random sample of the contextual features collected for the words in the query *[acid fast bacteria]*, and how the query's vector is generated by using the geometric mean of the frequencies of the features in the vectors for the query words.

4 Experiments and results

4.1 Experimental settings

To collect the contextual features for words and phrases, we have used a corpus of hundreds of millions of documents crawled from the Web in August 2008. An HTML parser is used to extract text and non-English documents are discarded. After process, the remaining corpus contains hundreds of billions of words.

As a source of keywords, we have used the top one and a half million English queries sent to the Google search engine after being fully anonymized. We have calculated the pairwise similarity between all queries, which would potentially return 2.25 trillion similarity scores, but in practice returns a much smaller number as many pairs have non-overlapping contexts.

As a baseline, we have used a new implementation of the Web Kernel similarity (Sahami and Heilman, 2006). The parameters are set the same as reported in the paper with the exception of the snippet size; in their study, the size was limited to 1,000 characters and in our system, the normal snippet returned by Google is used (around 160 characters).

In order to evaluate our system, we prepared a goldstandard set of query similarities. We have randomly sampled 65 queries from our full dataset, and obtained the top 20 suggestions from both the Sahami system and the distributional similarities system. Two human raters have rated the original query and the union of the sets of suggestions, using the same 5-point Likert scale that Sahami used. Table 2 shows the confusion matrix of scores between the two raters. Most of the disagreements are between the scores 0 and 1, which means that probably it was not clear enough whether the queries were unrelated or only slightly related. It is also noteworthy that in this case, very few rewritten queries were classified as being better than the original, which also suggests to us that probably we could remove the topmost score from the classifications scale.

We have evaluated inter-judge agreement in the following two ways: first, using the weighted Kappa score, which has a value of 0.7111. Second, by grouping the pairs judged as irrelevant or slightly relevant (scores 0 and 1) as a class containing negative examples, and the pairs judged as very relevant, equal or better (scores 2 through 4) as a class containing positive examples. Using this two-class clas-

Method	Prec@1	Prec@3	Prec@5	mAP	AUC
Web Kernel	0.39	0.35	0.32	0.49	0.22
Unigrams	0.47	0.53	0.47	0.57	0.26
N-grams	0.70	0.57	0.52	0.71	0.54

Table 3: Results. mAP is mean average precision, and AUC is the area under the precision/recall curve.

sification, Cohen's Kappa score becomes 0.6171. Both scores indicates substantial agreement amongst the raters.

The data set thus collected is a ranked list of suggestions for each query[1], and can be used to evaluate any other suggestion-ranking system.

4.2 Experiments and results

As an evolution of the distributional similarities approach, we also implemented a second version where the queries are chunked into phrases. The motivation for the second version is that, in some queries, like *[new york cheap hotel]*, it makes sense to handle *new york* as a single phrase with a single associated context vector collected from the web corpus. The list of valid n-grams is collected by combining several metrics, e.g. whether Wikipedia contains an entry with that name, or whether they appear quoted in query logs. The queries are then chunked greedily always preferring the longer n-gram from our list.

Table 3 shows the results of trying both systems on the same set of queries. The original system is the one called *Unigrams*, and the one that chunks the queries is the one called *N-grams*. The distributional similarity approaches outperform the web-based kernel on all the metrics, and chunking queries shows a good improvement over using unigrams.

5 Conclusions

This paper extends the vector-space model of distributional similarities to query-to-query similarities by combining different vectors using the geometric mean. We show that using n-grams to chunk the queries improves the results significantly. This outperforms the web-based kernel method, a state-of-the-art unsupervised query-to-query similarity technique, which is particularly relevant as the corpus-based method does not benefit automatically from search engine features.

[1]We plan to make it available to the research community.

References

S. Bordag. 2008. A Comparison of Co-occurrence and Similarity Measures as Simulations of Context. *Lecture Notes in Computer Science*, 4919:52.

J.R. Firth. 1957. A synopsis of linguistic theory 1930-1955. *Studies in Linguistic Analysis*, pages 1–32.

G. Grefenstette. 1992. Use of syntactic context to produce term association lists for text retrieval. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 89–97. ACM New York, NY, USA.

R. Jones, B. Rey, O. Madani, and W. Greiner. 2006. Generating query substitutions. In *Proceedings of the 15th international conference on World Wide Web*, pages 387–396. ACM New York, NY, USA.

Reiner Kraft and Jason Zien. 2004. Mining anchor text for query refinement. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 666–674, New York, NY, USA. ACM.

D. Lin. 1998. Extracting Collocations from Text Corpora. In *First Workshop on Computational Terminology*, pages 57–63.

Stefan Riezler, Yi Liu, and Alexander Vasserman. 2008. Translating Queries into Snippets for Improved Query Expansion. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*.

E. Riloff and J. Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 117–124. Association for Computational Linguistics.

M. Sahami and T.D. Heilman. 2006. A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the 15th international conference on World Wide Web*, pages 377–386.

G. Salton, A. Wong, and CS Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Egidio Terra and Charles L.A. Clarke. 2004. Scoring missing terms in information retrieval tasks. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 50–58, New York, NY, USA. ACM.

W. Yih and C. Meek. 2007. Improving Similarity Measures for Short Segments of Text. In *Proceedings of the Natural Conference on Artificial Intelligence*, volume 2, page 1489. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

W. Yih and C. Meek. 2008. Consistent Phrase Relevance Measures. *Data Mining and Audience Intelligence for Advertising (ADKDD 2008)*, page 37.