Unsupervised Query Categorization using Automatically-Built Concept Graphs

Eustache Diemert
Yahoo! - Search Innovation
Grenoble, France
eustache@yahoo-inc.com

Gilles Vandelle
Yahoo! - Search Innovation
Grenoble, France
gilles.vandelle@yahoo-inc.com

ABSTRACT

Automatic categorization of user queries is an important component of general purpose (Web) search engines, particularly for triggering rich, query-specific content and sponsored links. We propose an unsupervised learning scheme that reduces dramatically the cost of setting up and maintaining such a categorizer, while retaining good categorization power. The model is stored as a graph of concepts where graph edges represent the cross-reference between the concepts. Concepts and relations are extracted from query logs by an offline Web mining process, which uses a search engine as a powerful summarizer for building a concept graph. Empirical evaluation indicates that the system compares favorably on publicly available data sets (such as KDD Cup 2005) as well as on portions of the current query stream of Yahoo! Search, where it is already changing the experience of millions of Web search users.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Storage and Retrieval—*Information Search and Retrieval*

General Terms

Algorithms, Measurement, Performance, Experimentation

Keywords

query categorization, unsupervised learning, concept networks, cross-reference, knowledge based search, Web mining

1. INTRODUCTION

Web Search engines are nowadays an intrinsic part of daily life of hundreds of million of people; they are also part of a major industry of this century. Despite the massive user adoption of these services, the profitability of the companies that power them and the maturity of the search sciences that underpin the whole, there is considerable room left for improvement. Better understanding of user queries, and especially providing cheap, accurate topical categorization systems are necessary objectives for several reasons.

Firstly, end-users benefit from it, via the efficient use of the federated search paradigm, which will integrate and blend both traditional Web search results with so-called vertical search results, whether coming from open, structured databases like Wikipedia or from a federation of closed sources (e.g. partner content like specialized news feeds). This approach enables the display of rich content into the search result page. For example, this could mean for music related queries displaying artist news and videos, offering track samples or videos. From user surveys and analysis of click-through rates, we can deduce heightened user interest on such specialized query-specific content.

Secondly, search engine businesses and search advertisers also benefit from it because topical categorization can help displaying more relevant sponsored search adverts. With adequate categorization quality, the advertisers can then buy placements not directly attached to individual keywords, but rather to categories, thus enabling the leveraging of the long tail of rare queries [4].

Finally, general purpose query categorization can be seen as an enabler or facilitator for the design and implementation of additional advanced services, like blending or redirection between services and advanced query analysis.

The origin of this research is basically the need for an efficient way to implement a federated search paradigm into Web search, for a broad set of different markets. To pursue this goal, we assumed very limited editorial resources were available and potentially a large number of different languages and cultures were to be covered, eliminating from the start any solution whose success would depend too much on editorial analysis or costly linguistic data.

As a consequence, we explored the following major research questions:

- Can we design an unsupervised, yet efficient query categorization system ?
- Can we realistically implement such a system on a very large scale (all the Web being the target) ?
- Can we propose an innovative approach using only unstructured Web data to build the categorizer model ?

Traditional approaches include synonym based classifiers and supervised learning on (query, categories) pairs. The first type of systems was simply unacceptable due to the cost constraint of such approaches. The second type raised questions on the availability and cost of preparing training corpora for markets generating limited revenue, as is the case in many small countries. This lead us to consider the unusual and difficult option of designing a novel, unsupervised approach. However, a key constraint is that the categorization power must remain high, as errors in the selection of

WWW 2009, April 20–24, 2009, Madrid, Spain.
ACM 978-1-60558-487-4/09/04.

the vertical databases to query can have dramatic impact on the relevancy and user experience[1].

1.1 Contributions

The major contribution of this paper is reporting that an unsupervised learning paradigm can match the categorization power of supervised, costly approaches. To our knowledge, it is the first reported system to work without training instances, be it already known queries or documents. It must be further emphasized that we also prove in the evaluation that the proposed approach performs well on different taxonomies which differ by the granularity of their categories and without tuning any parameters, thus providing an off-the-shelf categorizer for different problems.

Subsequently, from a computational linguistics and machine learning point of view, we discovered that cross-reference concept graphs can summarize a massive amount of knowledge derived from unstructured data (query logs and Web documents) and, when analyzed with proper algorithms, provide a proper environment for query analysis.

In this paper we will describe our categorizer model (Section 2), evaluate it for query categorization (Section 3) and compare it against other published methods (Section 4).

2. KNOWLEDGE BASED SEARCH

The approach we present here is implemented as a part of what we call the Knowledge Based Search (KBS) platform. KBS has several other usages including query sense disambiguation, query rewriting and search suggestions, all of which utilize the concept graph. We will concentrate here on the aspects that play a role in query categorization. Firstly, we will focus on the concept graph model that drives the system. Then, we will fully describe the algorithms that implement query categorization in this setting.

2.1 Cross-reference Concept Graph Model

In our implementation, there is only one type of node, which corresponds to concepts. Concepts are represented as words or phrases. Edges bear a weight corresponding to the degree of cross-reference ($xref$) between the concepts. $xref$ is computed thanks to a search engine. Using standard Information Retrieval notation, we can define the raw $xref$ between two terms:

$$xref_{raw}(t, t') := \sum_{d \in D_t} TF(d, t') \quad (1)$$

where $TF(d, t')$ is the term frequency of term t' in document d and D_t is the set of top results from a search engine. Basically, $xref_{raw}$ is the sum of the term frequencies of term t' in the result set of term t. TF is usually normalized to document length and associated with an IDF term that scales down the importance of frequent terms [15]. As our concepts may be represented by single words or phrases, it is thus not straightforward to normalize (1). Using a non-normalized $xref$ would be inconvenient as (1) wouldn't be comparable between queries. Moreover, computing an inverse document frequency on phrases can be costly, depending on the search engine implementation. That's why we prefered to use a slightly different and simpler calculation which roughly estimates (1) while having the nice property of being normalized:

$$xref(t, t') \sim \frac{1}{|D_t|} \sum_{d \in D_t} \mathbf{1}_{[t' \in d]} \quad (2)$$

which is the ratio of documents that contain term t' in the result set of query t. This quantity has the advantage of being normalized by definition and straightforward to compute given access to a subset of the result set on any search engine, while capturing the intuitive sense of (1). Indeed, it can be viewed as a boolean model of (1) where each document confers only a binary weight to $xref$.

In fact, we omitted an important parameter in (2) : N, the number of results to take into account in D_t. This parameter has a serious impact on the values of $xref$. If we stick to $N = 10$ for example[2], we will only capture cross-references that are obvious. Conversely, setting $N = 40$ or more has the effect of capturing more relations between terms, but with the drawback of adding noise. Finally, Equation 3 fully defines the $xref$ that we use in our model :

$$xref_N(t, t') = \frac{1}{N} \sum_{d \in D_t^N} \mathbf{1}_{[t' \in d]} \quad (3)$$

We will assume from now on that N is set to a suitably low value that takes advantage of the ranking function of the search engine (the optimal value is likely to be implementation dependent, and thus less interesting in the context of this paper).

Using cross-reference as the primary metric to build a concept graph seems not to have been investigated before. Previous work on automatically integrating unstructured data to build concept graphs traditionally use various formulations of co-occurrence rather than cross-reference to link concepts together. The entity containment graph extracted from Wikipedia in [24] uses co-occurrence at the paragraph level to compute relationships between entities. Another example of co-occurrence graph can be found in [17] where the authors conduct a detailed analysis of the statistical mechanics of a graph extracted from the Reuters-21578 news corpus. Co-occurrence can be interpreted as the semantic proximity between concepts, whereas $xref$ encode a different relationship. We further investigate the properties of this relationship and its benefits for categorization.

A first advantage of $xref$ is that it allows to use Web query logs as a corpus to build the concept graph. In fact, the use of $xref$ is scalable with respect to the amount of data that we can process, since it requires only query level processing, a task for which search engines are typically optimized (e.g. cache infrastructure). Computing co-occurrence on a collection of the size of the web (or significant portions of it) would be prohibitive as it requires document level processing (for example counting the number of phrases in which two words occur together) for which search engines databases are not designed for.

[1] think, for example, of a database of songs. It is simple to find song titles that are also common words, so that if a query is misguided to the database, it is likely that it will return a hit. In the music feeds we worked on, titles like "Love","play", etc appeared several dozen of times; additionally passing on unnecessary queries incurs an additional cost in terms of vertical search indexes' load [6]

[2] as major Web search engines are usually tuned to give the best performance and relevancy on the first result page, that's to say on the 10 first documents

Using a search engine has an additional advantage : we benefit from the finely tuned relevance model implemented in the ranking function, as well as the different spam and undesirable content filtering.

We also put two additional constraints on the graph model. First, it must be a Directed Acyclic Graph (DAG). This is achieved by running a batch mode cycle remover process that removes only the weakest edge of a given cycle. The acyclicity property is important in terms of computational efficiency as it simplifies the implementation of the graph storage and retrieval primitives. It allows us not to bother implementing cycle detection in the graph traversal algorithms that we use at runtime.

Next, we only allow one edge between two given nodes. This means that when building the graph we retain only one $xref$ value between two nodes : even if $xref(t, t') \geq 0$ and $xref(t', t) \geq 0$ we will only create one edge between t and t' and give it an orientation corresponding to the heaviest $xref$. Namely, if $xref(t, t') > xref(t', t)$ then the edge will be oriented like this: $t \rightarrow t'$. This edge orientation constraint has a very important effect on the model, as we found it influences greatly the graph topology. At the statistical level, it modifies the out degree distribution by skewing the otherwise familiar power law distribution as can be seen in Figure 1. As a consequence, we experienced on a 350000 nodes graph a maximum in-degree of $\sim 2^{12}$ whereas maximum out-degree was no more than $\sim 2^8$.

We believe that these constraints enforce a graph topology that tends to favor paths leading to well connected nodes and limits the paths going out of these well connected nodes.

We will now focus on the interpretation of the $xref$ measure. Based on our experience with the concept graph we will describe three main properties of $xref$.

PROPERTY 1. *$xref$ encodes a Conceptual Genericity relation*

As genericity is a conceptual rather than computational notion [22], we assume there is no formal proof of this proposition. However, we believe anyone used to searching the Web has an intuitive comprehension of this property. For example, when querying for "www 2009", it is likely several top ranked documents will contain concepts such as "world wide web conference" and "Madrid", whereas the reverse $xref$ will be much lower, as "Madrid" is about so much other concepts than just "www 2009" (see Table 1).

From	To	$xref$
www 2009	Madrid	0.10
www 2009	world wide web conference	0.08
Madrid	www2009	0.00
Indiana Jones	Movies	0.28
Movies	Indiana Jones	0.00
Football	Sports	0.62
Sports	Football	0.78

Table 1: Examples of $xref$ values

Another empirical evidence of this property can be exhibited on movie names for example. It is likely that most movie names will have a strong $xref$ to some generic nodes like Movies, Cinema etc, whereas the reverse $xref$ will be probably very low[3].

When designing the whole system we made the assumption that this property holds in a vast majority of cases and will allow the system to provide good generalization power. Indeed, we can deduce that applying property 1 to all concepts will result in the ability to link any specific concept (e.g. concepts representing a given query) to more generic concepts (e.g. concepts representing categories). This is of the uttermost interest since it allows us to redefine the categorization task as generalizing the sense of a given query until it matches the sense of a category. This idea has been implemented in the algorithms described later in Section 2.3.

The scope of this property is limited by two sister properties:

PROPERTY 2. *The conceptual relation of genericity is web bound.*

Indeed, when looking at Table 1, one easily notices that the resulting edge between Football and Sports would be $Sports \rightarrow Football$, which conflicts with the orientation most humans would choose consciously when presented with both alternatives. Actually, it may be seen as an oddity, but we tend to think it is an important feature of the system because it reflects the reality of the web corpus. In that sense it is a very different resource than editorially based taxonomies and ontologies. Another example is the query "rose" : an ontology based system or a concept graph extracted from a non-web corpus will probably answer that the two main senses of the query are referring to a color and a flower. On a web search engine, you tend to have different meanings : the flower and the rock band "Guns'n'Roses", due to the popularity of these two concepts on the web.

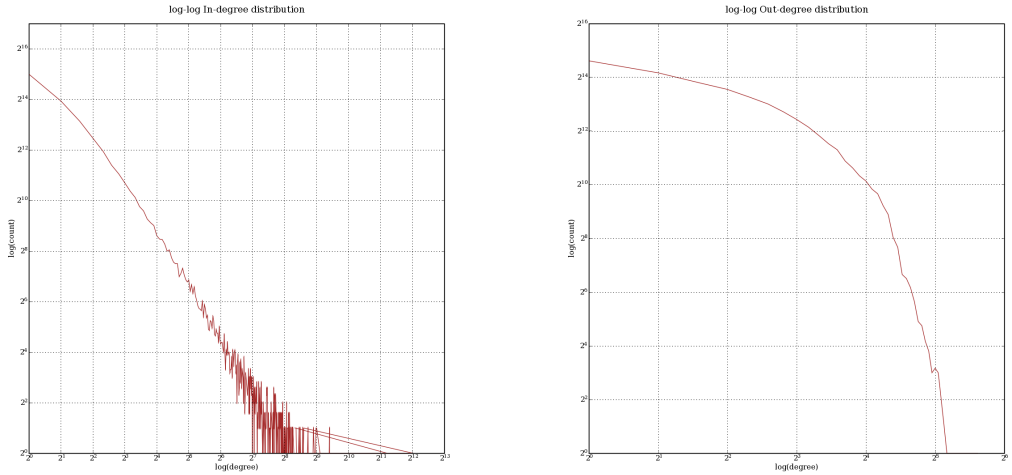
PROPERTY 3. *The conceptual relation of genericity is weakly transitive.*

At the graph level, when following a path, the weight of each edge plays an important role in making the genericity relation hold or not to the next node. Of course, a weak edge in a given path downgrades the genericity relation between the start and end nodes of the path. But if the path is too long, the transitivity won't hold either. A funny evidence is that we found a path starting from "Madonna" and leading to both "A. Einstein" and "Nuclear Physics" in no more than 7 edges. When checking the different weights along the path, some were weak, but none of them was lower than 0.1. This property had a deep influence on the way we designed the algorithms that use the concept graph for categorization, as we will see later.

2.2 Digression : Concepts

In the previous sections, we described our model as a concept graph but we didn't properly define what type of concepts we refer to, nor did we describe the way they are extracted. Basically, we defined $xref$ on terms, without putting any constraint on the nature of the terms. It would thus seem reasonable to compute $xref$ on whole phrases as well as on single words. As other studies [8][21] suggested that noun-phrases are easy to extract and that they are good

[3]unless the N parameter is set to a value large enough compared to the size of the collection (in our case : the Web!)



(a) in-degree (b) out-degree

Figure 1: $log_2 - log_2$ plot of degree distribution of a 350k nodes $xref$ Concept Graph

at capturing the sense of queries in the context of information retrieval, we use such concepts in our implementation. Namely, we use *prisma* [1], a text mining technology that detects noun-phrases in text corpora. Indeed, all nodes in our graph are "prisma concepts". However, we do believe that any other concept or entity extraction technique could be used without any other loss in quality than the one induced by the precision of the concept extraction. In our study, we set the system to extract the top-20 most salient concepts for each document.

2.3 Query Categorization with the Concept Graph

To leverage the concept graph for query categorization, we propose to focus on relevant portions of the graph at query-time. These will then be mined to discover the nodes that best describe the query. Finally, multiple methods could then be used to categorize the query using the most relevant nodes detected.

2.3.1 Hook-Categories Algorithm

In the KBS system, we chose to directly integrate the categories as special nodes in the graph. This approach has the advantage of providing very simple and dynamic means of maintaining the categorizer. Adding, deleting or changing the scope of a category means then only adding, deleting or changing the linkage of the category node. We use the Hook-Category (Algorithm 1) to integrate a given category into the graph.

The algorithm basically detects *descriptors* : nodes that best describe the target category. These will be used in the query categorization algorithm. The input category can be used alone, but it usually helps if some keywords (what we call *seeds*) describing the category are available. For the federated search use case, we took at most 10 terms from the keywords that appear on the summary of the search result page when typing the label of the vertical. For instance, for the "Health" vertical, we used {diet, fitness, longevity, disease, symptoms, treatments} as input because they were

Algorithm 1 Hook-Category

INPUT: a graph $G(E, V)$ + a list of seeds + a threshold δ
OUTPUT: a list of descriptors for the category

```
descriptors ← {}
for all keyword in seeds(category) do
  node ← match(keyword)
  if node ≠ null then
    descriptors ← descriptors ∪ node
    for all neighbor in succ(node) ∪ pred(node) do
      if xref(node, neighbor) ≥ δ and
      xref(neighbor, node) ≥ δ then
        descriptors ← descriptors ∪ neighbor
      end if
    end for
  end if
end for
```

part of the snippet of the first result page when running the query "health". For each seed, the algorithm seeks a corresponding node in the graph using a basic string matching primitive. Each of these are promoted to descriptors. Then, the algorithm tries to find other strongly related nodes in the neighborhood of each descriptor. Those matching the condition will be promoted to descriptors too. The threshold δ was determined after manual reviews (we found that $\delta = 0.5$ produces a very small topical drift) and is mainly implementation dependent as it explicitly controls $xref$ which in turn can be different from one search engine to another.

Table 2 shows a typical run of the Hook-Category Algorithm.

Having identified the *descriptors*, the category is then plugged into the graph by creating full strength links from all the descriptors. To distinguish from other nodes, the type is set to "domain". This may be seen in Figure 2, where the "Movies" domain is boxed and labeled "domain", receiv-

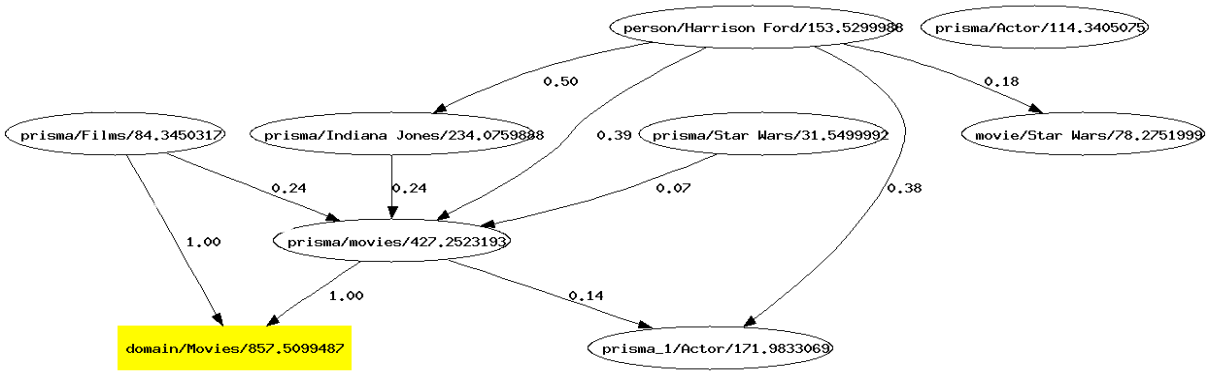


Figure 2: Sample concept graph for the query "indiana jones". Category nodes are boxed. Each node is labeled with a type/value/weight triplet, the weight being produced by the Categorize algorithm. Edges are labeled with the $xref$ value.

Input (*seeds*)	movies, cinema
Output (*descriptors*)	movies, cinema, films, actor, dvd, showtimes, trailers, imdb

Table 2: Sample results of the Hook-Category Algorithm

ing two incoming edges from "Films" and "Movies", whereas other nodes are labeled "prisma".

2.3.2 Categorization Algorithm

This is the very core of the KBS query categorizer. Given a query at runtime, we take the most salient concepts of the N first documents in the result page and weight them with their frequency of occurrence in these documents according to Equation 4:

$$weight_{initial}(concept) := \frac{1}{N} \sum_{d \in D_q^N} \mathbf{1}_{[concept \in d]} \qquad (4)$$

The number of documents to consider (N) might not be the same as in Equation 3. In our implementation we set $N = 10$ for performance reasons. In an industrial implementation, we can reasonably expect the documents to be indexed with their top concepts.

The $(concept, weight_{initial})$ pairs are then used as input for the Categorize algorithm (Algorithm 2) to find the best category related to the query, if any.

This algorithm is similar to many graph mining algorithms that use random walks to detect salient nodes, except that it is limited to a fairly reduced number of steps. We found reasonable values for N_{iter} to be around 4. Above this value we experienced a topical drift due to the weak transitivity of $xref$ as explained in the previous sections. Even if the weight propagation is limited in essence by $xref$ being a normalized value in the $[0, 1]$ range, we believe it is not reasonable to go further.

An example of the result of the Categorize algorithm can be viewed in Figure 3. In this case, the query "spurs" has two possible categories: "Football" and "Basketball"[4]. According to KBS, the best is "Basketball" with a weight of 44.6 versus 26.2 for "Football". These weights can then be interpreted as a degree of relation between the query and each domain. We can also choose to keep only the heaviest in case we only want boolean categorization. The score is also important to derive the confidence with which the system did the prediction.

[4]"spurs" can refer to the "Tottenham Hotspurs" football team or to the "San Antonio Spurs" basketball team

Algorithm 2 Categorize

INPUT: a graph $G(E, V)$ + a list of (terms,weight) pairs + a free parameter N_{iter}
OUTPUT: a weighted category list

```
for all v in V do
  weights[node] ← 0.0
end for
for all (term, weight) in input do
  node ← match(term)
  if node ≠ null then
    weights[node] ← weight
  end if
end for
for i = 0 to N_iter do
  for all v in V do
    for all p in pred(v) do
      weights[v] ← weights[v] + weights[p] * xref(p, v)
    end for
  end for
end for
return {domain ∈ G} weighted by weights[domain]
```

3. EVALUATION

In this section we evaluate our unsupervised query categorization system. First we will give some details regarding the way we built the model. Then we will report on relevancy evaluations. Finally, we will assess the coverage on the KBS categorizer on a real world query stream.

For the purpose of this study, we built a concept graph using 2 million queries randomly sampled from Yahoo! Search as a raw input. These queries were processed using the *prisma* technology to extract the concepts in them. After this step, we expanded our concept list by querying the search engine, once per concept, in order to retrieve addi-

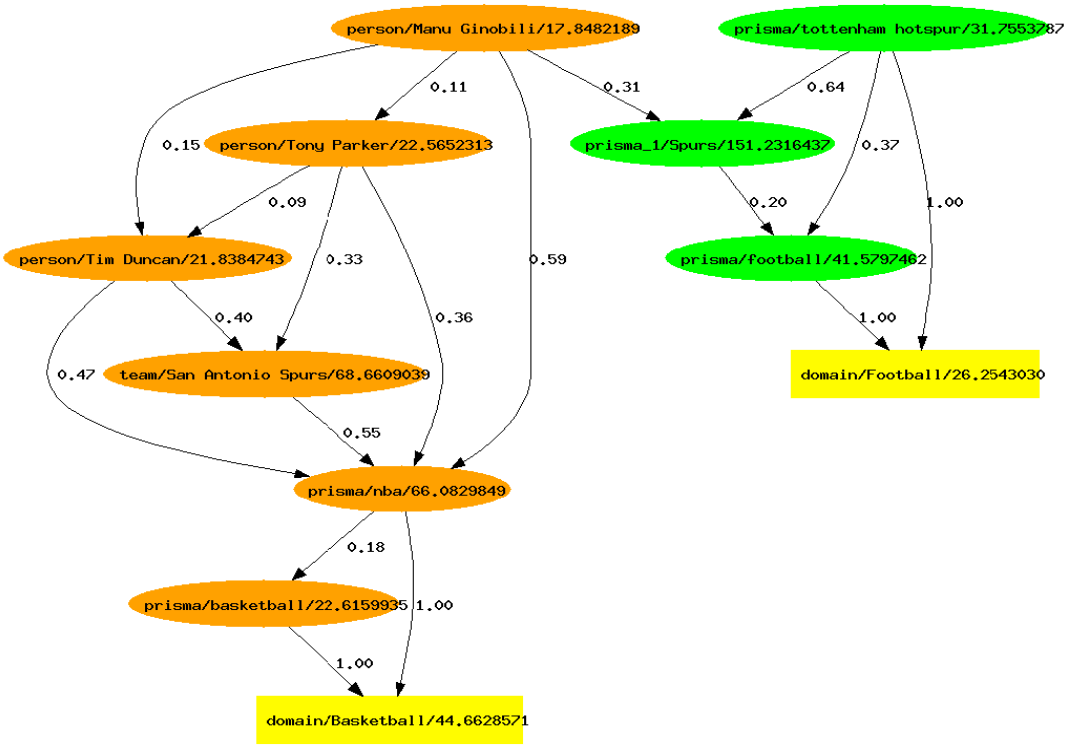


Figure 3: Sample concept graph for the query "spurs". This query is ambiguous between the Basketball and Football domains. Some nodes are typed as "person" or "team" as a by-product of the concept extraction, which can be ignored in this discussion.

tional ones. This step multiplied by ~ 3 the number of unique concepts in our list. After these steps, we computed $xref$ for all the concept pairs. The whole process produced a graph of 2.30 million nodes and 6.74 million edges.

For relevancy evaluation, we report results on two different datasets. The first one aims at testing KBS for the federated search use case. The second one is the KDD Cup 2005 query classification task against which many proposed approaches have already been evaluated. We give standard metrics for both evaluations : precision, recall and F_1 for comparison with other published studies. KBS was used as a binary categorizer in both experiments, retaining only the heaviest category output for each query.

3.1 Editorial evaluation for Federated Search

This dataset was extracted from random queries sampled from Yahoo! Search US query logs in February 2007. Human evaluators removed navigational (mainly URLs or website names) and transactional (e.g. shopping) queries, resulting in a reduced set of 3151 queries. Professional editors were provided with a taxonomy that consisted of 9 categories corresponding to vertical, production search systems : Music, Travel, Finance, Movies, Jobs, Health, Games, Sports, Autos. They manually classified each query into at most one category. This process showed that a variety of queries representing 26% of the total couldn't be mapped to the given taxonomy.

The intended usage is that once a prediction is made for a given query, one of these vertical search systems is then queried for fresher, more focused and richer content than what can be found in the general index. Thus, precision is of great importance for this application, as false positives might put undue stress on these systems, as well as display irrelevant elements on the search result page.

We compare our results to a baseline classifier that uses only word features. It is based on Support Vector Machines (SVM) [10] and trained on half of the dataset using a radial basis function as kernel. The training phase included optimization of the γ kernel parameter and of the error penalty. Testing was done on the other half of the corpus using a one-against-all voting strategy. As SVMs are among the best text classifiers to date [5][15], the baseline outlines what can be achieved using only word features from the query.

Results are as follows :

Metrics	Baseline	KBS	Diff. (%)
micro averaged Recall	0.503	0.627	+24.6
micro averaged Precision	0.491	0.819	+66.8
micro averaged F_1	0.497	0.710	+42.8

Table 3: Global Relevancy Metrics on 3k random queries from Yahoo! Search US

Our system shows an improvement in F_1 of 43% compared to state of the art methods for text categorization.

KBS' overall precision is above 0.80, which is an excellent score when compared to usually reported figures (see for instance the KDD Cup evaluation where the most precise system yield around 0.75 precision), especially given the good level of recall : 0.63. Please note again that these results were produced by an unsupervised learning scheme (thus not involving any learning on the queries themselves).

When analyzing the failure patterns in this test, we noticed that the worst confusion between categories was 3.9% between Sports and Autos. As the Sports category includes all motor sports, this confusion seems reasonable. Generic false negative patterns included (in order of importance) :

- queries returning no results (including misspellings not corrected by the search engine. E.g. "chrisbenoi wickkepida").
- noise introduced during the extraction of concepts : sometimes the concepts returned are irrelevant or poorly weighted, one anecdotal concept being over-weighted.
- ambiguous queries for which KBS didn't choose the same best category as the editors. E.g. "radiology jobs" for which KBS ranked Health first, then Jobs.

Category	Recall	Precision
Music	0.758	0.854
Travel	0.691	0.919
Finance	0.683	0.785
Autos	0.854	0.928
Movies	0.803	0.688
Jobs	0.661	0.905
Health	0.817	0.838
Games	0.744	0.895
Sports	0.773	0.791

Table 4: Detailed Relevancy Metrics

Table 4 presents category-wise metrics. When inspecting these metrics, please note that the lowest precision is above 0.68, an important fact for the federated search application.

3.2 KDD Cup 2005

Our second evaluation uses the KDD Cup 2005 dataset [13]. This dataset consists of 800 evaluated queries extracted from MSN Search in 2005 and manually classified into 67 categories by 3 human evaluators. Several other papers between 2005 and today used it as a comparison point [20] [2]. We didn't tune the system in any way for the KDD Cup categories and thus evaluated KBS as an off-the-shelf categorizer. Indeed, we fed the Hook-Categories algorithm with only the category names split on ampersands and slashes (e.g. Shopping/Bargains & Discounts $\mapsto$ shopping, bargains, discounts), thus avoiding any third party keyword generation method and associated additional bias.

Best reported figures during the competition and afterward are reported in Table 5, along with KBS performance for this evaluation.

System	Precision	F_1	Recall
KDD Cup Winner [18]	0.754	0.444	N/A
Best today [20]	0.828	0.461	N/A
KBS	0.614	0.460	0.368

Table 5: Micro-averaged Relevancy Metrics on KDD Cup 2005.

Again, we report an outstanding F_1 score, better than all systems that were engaged in the competition and matching the quality of the best reported approach as of today [20].

Please note again that KBS is subject to constraints unknown to all other reported approaches. Namely, the system doesn't make use of the training corpus furnished with the dataset (1200 queries) and wasn't trained in whatsoever manner on the test queries. Compared to the best reported approach, which uses third-party taxonomies, we emphasize again that our model doesn't make use of any external resource other than the search engine and its query logs.

In this evaluation, the number of categories makes the task harder for an unsupervised learning algorithm like KBS than for a supervised learning algorithm since the last one can often benefit from very precise information on category separation.

Failure patterns we discovered are as follows:

- failure to find any matching nodes during Hook-Categories (20% of the queries)
- failure to extract relevant concepts from the search engine results (8.7% of the queries)

The first failure pattern is, of course, mainly due to the fact that we only used the category names as seeds and could be easily overcome by using a third party keyword expander. One possible approach might be to leverage an existing document level taxonomy to extract additional concepts describing the categories. But this option does not fit in the scope of this experiment as we wanted to evaluate KBS in the purest setting possible. The second one is mostly due to the difficulty of the task of finding relevant concepts from documents in the result set. Some categories were severely penalized by this pattern: Information/Local & Regional, Online Community/Homepages, Shopping/Bargains & Discounts, Online Community/People Search to cite a few. The root cause is that these categories embody implicit rather than explicit characteristics. For instance, very few (if any) web documents belonging to the Local & Regional categories have concepts in it saying it is local. The concepts found were rather the names of the town or county and even if it would be possible, the Hook-Categories algorithm is a fairly bad solution to link all possible localities concepts to the category node.

3.3 Coverage on a real world query stream

To further assess the capability of KBS to provide query categorization on large portions of real world query streams, we evaluated its coverage (i.e. the ratio of queries on which KBS predicted that they belong to at least one category) using the KDD Cup categories on 500000 queries randomly sampled from Yahoo! Search UK in September 2008. 81.01% of the queries in this data appeared only once, thus assessing the presence of long tail queries.

Table 6 show the categories that show up on more than 2% of the queries along with their frequencies.

We found out that 68.2% of the queries did trigger a category prediction. This means that a fairly large portion of the query stream (including many rare queries) could potentially benefit from content not present in the main index. We also expect the coverage to raise if we augment the size of the concept graph, for example by including concepts found in documents retrieved for uncovered queries. Actually, the production version of KBS has this learning feedback enabled by default.

Category	Frequency (%)
Living/Career & Jobs	4.786
Entertainment/Movies	4.782
Entertainment/Games & Toys	4.204
Information/Law & Politics	3.964
Living/Fashion & Apparel	2.880
Information/Arts & Humanities	2.876
Living/Real Estate	2.846
Entertainment/Celebrities	2.792
Information/Education	2.720
Living/Travel & Vacation	2.712
Living/Pets & Animals	2.612
Living/Health & Fitness	2.436
Entertainment/TV	2.408
Shopping/Bargains & Discounts	2.126
...	...
Total	68.180

Table 6: Frequency of KDD Cup Categories on 500k random queries from Yahoo! Search UK (excerpt)

4. RELATED WORK

Query categorization is one of the major branches of classification on the trunk of modern information retrieval. While sharing a common background with document classification (see for example the work in [7] where the authors explore the classification of Web documents into taxonomies), it has to deal with very specific constraints unknown to other domains. The major constraint is of course the length of the queries, which is 2-3 words on average, thus reducing the number of word features available to classifiers. This is seen as one of the main reasons that explain the difficulty of the task.

Another aspect of this task is that the categories used should help understanding the user intent when typing the query. Actually, the topical classification, as treated in this paper, is just one dimension of the task. Indeed, if we take the transactional, navigational, informational classes proposed by [3], these are orthogonal to the topical classes. This is also the case for the "localness" in [9] or the task classes in [11].

Focusing on recent work on topical classification, we can identify the following trends :

- *few available datasets*: to the best of our knowledge, there is only one publicly available, real world dataset for the Web queries categorization task : the KDD Cup 2005 described in [13]. It consists roughly of a 67 class taxonomy with 800 evaluated queries. We can only regret that the community doesn't have larger, more varied and recent benchmarks (4 years is a very long time period for rapidly evolving environments like the Web).

- *using external knowledge*: the winners and runners up [23][12][18][19] of the competition all agree on the importance of using external knowledge, an aspect also assessed by more recent works in their particular context [4][2]. We can only confirm this point of view as the concept graph built with query logs is the key part of our system.

- *predominance of supervised learning*: another major point of agreement is the use of supervised learning, be it as a sole learning mechanism [4][23] or combined with synonyms classifiers and exact look-up on dictionaries [2][18]. As stated in the introduction, dictionaries are costly resources that one can't necessarily afford for specific languages and cultures or markets. In a sense, one might consider the Hook-Category algorithm as a far cousin of a synonym based classifier. The major advantage of KBS compared to synonym based classifiers is that categories hooked in the graph can receive weight from a large variety of more specific concepts which results in a better generalization power.

- *using document level classifiers as an intermediary step*: Some of the KDD Cup participants, as well as more recent work [23][20] also rely on the availability of document level taxonomies that can be mapped to the target classes. We can only stress the originality and advantage of our system : its independence from any linguistic or editorial resource, like existing training sets at the query or document level.

- *search engine as a mandatory building block*: in the setting of a commercial search engine, the only resource you can afford "for free" is the search engine itself. But even without considering resource constraints, quite all the reported systems use a search engine at one step or another. As our own system relies heavily on Yahoo! Search Technology, we believe that search engines are nowadays mandatory building blocks of a vast majority of advanced services and applications.

The best system engaged in the KDD Cup challenge [18] is based on the fusion of synonym-based and statistical, supervised classifiers plugged on 3 different search engines. The statistical classifiers rely on the existence of document-level training data. No evidence is given in the paper that the system could run with more limited resources, nor that the ensemble classifiers approach could be run at query time (that's to say with very tight time constraints). The advantage of our approach is that we have proposed an efficient, scalable way of computing the model and a simple, efficient categorization algorithm that can be implemented in the runtime part of a search engine.

A recent study [4] reports the most impressive figures so far on a non-disclosable dataset of rare queries. The proposed approach is based on a document-level, supervised classifier that is run on the top results returned by a search engine. While being similar in its use of the search engine, KBS has the advantage of being unsupervised and consequently to be independent of any training data.

5. CONCLUSION

In this paper we presented a novel, unsupervised, yet efficient approach for query categorization based on an automatically built concept graph. We explored the properties of cross-reference as a powerful conceptual generalization method. We evaluated the approach against both in-house and publicly available datasets, reporting metrics that show that KBS matches or outperforms other reported approaches on the given datasets. We noted the quality of these results

and the fact that they were produced by a fully automatic, unsupervised, low cost and language agnostic system.

As a by-product, we can also report that a version of KBS has been successfully deployed in production on Yahoo! Search UK as a Federated Search enabler. Indeed, each query made on the UK site triggers the KBS query categorizer, producing load peaks of 200+ queries per second with an average response time under 20 milliseconds. This is maybe the clearest demonstration of the efficiency of our approach. Practically speaking, it means that the query execution plan is modified according to the results of the categorizer to display different page layouts and rich content to the user. Click-through rates analysis showed that the user engagement is good on these new features.

From a network sciences point of view, providing additional insights into the mechanics of the concept graph is part of our future investigations. For instance, we would like to better understand the structure of cross-reference graphs, to which extent their topology can be constrained by the "one-edge" rule and which properties can be deduced from it.

Also, beyond the utility of the conceptual genericity property, finding a way of reasoning on the concept graph as was proposed in the semantic networks world [16][14] is a challenging research direction. We think it would be a much valued advance in future research if an approach could be proposed that combines ontologies or other semantic sources with a system like KBS that gives conceptual insights. For example, consider the problem of using ontologies for information retrieval. We might be interested, for instance, in the disambiguation of queries. When an ontology detects that a given word is ambiguous, it is usually unable to *rank* the different meanings. In this setting, we believe KBS could be of great help (see the "rose" case in section 2.1 for example), as it sums up knowledge found on the Web in the concept graph.

From an application point of view, we believe the concept graph is a powerful resource to be used for higher tasks in the field of Web information retrieval, for example in advanced analysis of queries, or in the ranking of query intents. Indeed, it is likely that the future of Web search will be centered around the detection of user intent and a better comprehension of the task she is involved in. An enlightening example is the query "blue suede shoes"[5]. Without a proper query categorization solution, many intent detection systems would be eventually classifying it as a transactional/shopping intent, whereas KBS is already capable of adding a level of conceptual context by tagging it as belonging to the "Music" category. If properly integrated into such a system, KBS would provide useful insight for the query intent detection task at a low cost and for all markets.

Finally, we conjecture that the concept graph we have described is an instance of a larger class of dimensional reduction techniques. We lack space to fully develop our argument, but the fact that the concept graph organizes the concepts from most specific to most generic can be seen as relatively similar to the principle of Principal Components Analysis, a technique traditionally used in statistical learning to reduce the number of dimensions in the dataset. Indeed, the concept graph can be seen as the weighted result of a form of hierarchical clustering. Each node in the graph can then be viewed as a summary of its predecessors. In the field of Web information retrieval, where the traditional vector space model [15] often counts several millions of dimensions, we believe such a dimensional reduction technique could be very useful to reduce the complexity of a number of tasks like document clustering and classification for example.

Giving a formal proof of this conjecture and exploring the theoretical questions around it will open perspectives of future research.

[5]which is the name of a song by Elvis Presley

6. REFERENCES

[1] P. Anick, V. Murthi, and S. Sebastian. Similar term discovery using web search. In European, editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, 2008.

[2] S. M. Beitzel, E. C. Jensen, O. Frieder, D. Grossman, D. D. Lewis, A. Chowdhury, and A. Kolcz. Automatic web query classification using labeled and unlabeled training data. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 581–582, New York, NY, USA, 2005. ACM.

[3] A. Broder. A taxonomy of web search. *SIGIR Forum*, 36(2):3–10, 2002.

[4] A. Z. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using web knowledge. In *SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 231–238, New York, NY, USA, 2007. ACM.

[5] S. Chakrabarti. *Mining the Web: Analysis of Hypertext and Semi Structured Data*. Morgan Kaufmann, August 2002.

[6] A. Chowdhury and G. Pass. Operational requirements for scalable search systems. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 435–442, New York, NY, USA, 2003. ACM Press.

[7] S. Dumais and H. Chen. Hierarchical classification of web content. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263, New York, NY, USA, 2000. ACM Press.

[8] M. Géry, M. Hatem, and H. D. Vaufreydaz. Web as huge information source for noun phrases integration in the information retrieval process. In *IKE '02: International Conference on Information and Knowledge Engineering*, pages 72–77. CSREA Press, 2002.

[9] L. Gravano, V. Hatzivassiloglou, and R. Lichtenstein. Categorizing web queries according to geographical locality. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 325–333, New York, NY, USA, 2003. ACM.

[10] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In C. Nédellec and C. Rouveirol, editors, *Proceedings of*

ECML-98, 10th European Conference on Machine Learning, number 1398, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE.

[11] I.-H. Kang and G. Kim. Query type classification for web document retrieval. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 64–71, New York, NY, USA, 2003. ACM Press.

[12] Z. T. Kardkovács, D. Tikk, and Z. Bánsághi. The ferrety algorithm for the kdd cup 2005 problem. *SIGKDD Explor. Newsl.*, 7(2):111–116, 2005.

[13] Y. Li, Z. Zheng, and H. . Dai. Kdd cup-2005 report: facing a great challenge. *SIGKDD Explor. Newsl.*, 7(2):91–99, December 2005.

[14] S. Liu, F. Liu, C. Yu, and W. Meng. An effective approach to document retrieval via utilizing wordnet and recognizing phrases. In *SIGIR '04: Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 266–272, New York, NY, USA, 2004. ACM Press.

[15] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, July 2008.

[16] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. Wordnet: An on-line lexical database. *International Journal of Lexicography*, 3:235–244, 1990.

[17] A. Ozgur, B. Cetin, and H. Bingol. Co-occurrence network of reuters news. *International Journal of Modern Physics C*, 19(5):689–702, Dec 2008.

[18] D. Shen, R. Pan, J. T. Sun, J. J. Pan, K. Wu, J. Yin, and Q. Yang. Q2c@ust: our winning solution to query classification in kddcup 2005. *SIGKDD Explor. Newsl.*, 7(2):100–110, 2005.

[19] D. Shen, R. Pan, J. T. Sun, J. J. Pan, K. Wu, J. Yin, and Q. Yang. Query enrichment for web-query classification. *ACM Trans. Inf. Syst.*, 24(3):320–352, 2006.

[20] D. Shen, J. T. Sun, Q. Yang, and Z. Chen. Building bridges for web query classification. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 131–138, New York, NY, USA, 2006. ACM.

[21] Vechtomova and Olga. Noun phrases in interactive query expansion and document ranking. *Information Retrieval*, 9(4):399–420, September 2006.

[22] C. Vogel and M. Mcgillion. Genericity is conceptual, not semantic. Technical report, University of Dublin, Trinity College, July 2002.

[23] D. Vogel, S. Bickel, P. Haider, R. Schimpfky, P. Siemen, S. Bridges, and T. Scheffer. Classifying search engine queries using the web as background knowledge. *SIGKDD Explor. Newsl.*, 7(2):117–122, 2005.

[24] H. Zaragoza, H. Rode, P. Mika, J. Atserias, M. Ciaramita, and G. Attardi. Ranking very many typed entities on wikipedia. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 1015–1018, New York, NY, USA, 2007. ACM.