Automatic Extraction of Useful Facet Hierarchies from Text Databases

Wisam Dakka #1, Panagiotis G. Ipeirotis *2

#*Computer Science Department, Columbia University*
1214 Amsterdam Avenue, New York, NY 10027, USA
1`wisam@cs.columbia.edu`

**Department of Information, Operations, and Management Sciences, New York University*
44 West 4th Street, New York, NY 10012, USA
2`panos@nyu.edu`

Abstract— **Databases of text and text-annotated data constitute a significant fraction of the information available in electronic form. Searching and browsing are the typical ways that users locate items of interest in such databases. Faceted interfaces represent a new powerful paradigm that proved to be a successful complement to keyword searching. Thus far, the identification of the facets was either a manual procedure, or relied on *apriori* knowledge of the facets that can potentially appear in the underlying collection. In this paper, we present an *unsupervised* technique for automatic extraction of facets useful for browsing text databases. In particular, we observe, through a pilot study, that facet terms rarely appear in text documents, showing that we need external resources to identify useful facet terms. For this, we first identify important phrases in each document. Then, we expand each phrase with "context" phrases using external resources, such as WordNet and Wikipedia, causing facet terms to appear in the expanded database. Finally, we compare the term distributions in the original database and the expanded database to identify the terms that can be used to construct browsing facets. Our extensive user studies, using the Amazon Mechanical Turk service, show that our techniques produce facets with high precision and recall that are superior to existing approaches and help users locate interesting items faster.**

I. Introduction

Many web sites (such as YouTube, The New York Times, eBay, and Google Base) function on top of large databases and offer a variety of services. YouTube, for example, lets users share video segments; The New York Times archive offers access to articles published since 1851; eBay offers a database of products for sale; and users of Google Base can access a wide variety of items, such as recipes, job offers, resumes, products for sale, or even quotations.

These web sites provide various access mechanisms to help users find objects of interest. Searching is probably the most common mechanism. For example, YouTube users seeking particular video segments issue keyword queries to find the YouTube objects, which have previously been annotated with descriptive terms or full sentences. Searching is also used as the primary access method for many text databases, such as The New York Times archive. Users access articles from the archive using a search interface that allows them to narrow down their searches based on titles, author names, and specific time ranges. Searching has, in fact, often been the method of choice to access databases of textual and text-annotated objects.

Despite its simplicity, searching is not always the only desirable method for accessing a large database. Often, other access methods are necessary or preferred. For example, we often do not go to a movie rental store or bookstore only to rent or buy items we have in mind but also to explore and discover new items that may interest us. Both curious users and users with little knowledge of the content of the database are usually in need of discovering the underlying structure and content of the databases to find new items. For such scenarios, users cannot rely on search alone. In fact, ranking is not feasible in these scenarios because of the absence of a concrete user query, and every database item is a candidate of interest to the curious or unfamiliar users.

In order to support such exploratory interactions, the majority of the web sites mentioned above use a form of *concept hierarchies* to support browsing on top of large sets of items. Commonly, browsing is supported by a single hierarchy or a taxonomy that organizes thematically the contents of the database. Unfortunately, a single hierarchy can very rarely organize coherently the contents of a database. For example, consider an image database. Some users might want to browse by style, while other users might want to browse by topic. In a more general setting, users can utilize multiple independent *facets* for searching and browsing a database. In his *colon classification* in the early 1930s, the librarian Shiyali Ramamrita Ranganathan introduced the term *facet* into classification theory as "*a clearly defined, mutually exclusive, and collectively exhaustive aspect, property, or characteristic of a class or specific subject*" [1]. As an example, consider browsing a schedule of TV programs, by time, TV channel, title, or actor, among many other possible dimensions.

Early work by Pollitt [2] and, more recently, by Yee et al. [3] showed that *faceted hierarchies*, which allow users to browse across multiple dimensions, each associated with independent hierarchies, are superior to single monolithic hierarchies. For example, researching the New York Times archive can be enhanced by taking advantage of the *topic*, *time*, *location*, and *people* facets. Users can navigate within and between the independent hierarchies of the four facets. A faceted interface can be perceived as an OLAP-style cube over the text documents [4], which exposes the contents of the underlying database and can help users more quickly locate items of interest.

One of the bottlenecks in the deployment of faceted interfaces over databases of text or text-annotated documents is the need to manually identify useful dimensions or facets for browsing a database or lengthy search results. Once the facets are identified, a hierarchy is built and populated with the database items to enable the user to locate the items of interest through the hierarchy. Static, predefined facets and their manually or semi-manually constructed hierarchies are usually used. However, to allow wide deployment of faceted interfaces, we need to build techniques for *automatic construction of faceted interfaces*. Building a faceted interface on top of a database consists of two main steps:

- Identifying the facets that are useful for browsing the underlying database, and
- Building a hierarchy for each of the identified facets.

In this paper, we present an *unsupervised* technique that fully automates the extraction of useful facets from free-text. The basic idea behind our approach is that high-level facet terms rarely appear in the documents. For example, consider the named entity "*Jacques Chirac*." This term would appear under the facet "*People* → *Political Leaders*." Furthermore, this named entity also implies that the document can be potentially classified under the facet "*Regional* → *Europe*→ *France*." Unfortunately, these (facet) terms are not guaranteed to appear in the original text document. However, if we "expand" the named entity "*Jacques Chirac*" using an external resource, such as Wikipedia, we can expect to encounter these important "*context terms*" more frequently. Our hypothesis is that facet terms emerge after the expansion, and their frequency rank increases in the new, expanded database. In particular, we take advantage of this property of facet terms to automatically discover, in an unsupervised manner, a set of candidate facet terms from news articles. We can then automatically group together facet terms that belong to the same facet using a hierarchy construction algorithm [5] and build the appropriate browsing structure for each facet using our algorithm for the construction of faceted interfaces. In summary, the main contributions of our work are as follows:

- A technique to identify the important terms in a text document using Wikipedia,
- A technique that uses multiple external resources to identify facet terms that do not appear in the document but are useful for browsing a large document database, and
- An extensive experimental evaluation of our techniques that includes extensive user studies, which use the Amazon Mechanical Turk service for evaluating the quality and usefulness of the generated facets.

The rest of the paper is structured as follows. Section II gives the necessary background. Section III discusses the setup and the results of our pilot study with human subjects. Then, Section IV discusses our unsupervised techniques to identify facet terms, and Section V reports the results of our experimental evaluation. Finally, Section VI reviews related work, and Section VII concludes the paper.

II. Background

While work on the automatic construction of *faceted* interfaces is relatively new, automatic creation of subject hierarchies has been attracting interest for a long time, mainly in the form of *clustering* [6–8]. However, automatic clustering techniques generate clusters that are typically labeled using a set of keywords, resulting in category titles such as "*battery california technology mile state recharge impact official hour cost government*" [9]. While it is possible to understand the content of the documents in the cluster from the keywords, this presentation is hardly ideal.

An alternative to clustering is to generate hierarchies of *terms* for browsing the database. Sanderson and Croft [10] introduced the subsumption hierarchies and Lawrie and Croft [11] showed experimentally that subsumption hierarchies outperform lexical hierarchies [12–14]. Kominek and Kazman [15] used the hierarchical structure of WordNet [16] to offer a hierarchy view over the topics covered in videoconference discussions. Stoica and Hearst [17] also used WordNet together with a tree-minimization algorithm to create an appropriate concept hierarchy for a database. Recently, Snow et al. [5] showed how to improve the WordNet subsumption hierarchies by using evidence from multiple sources.

All these techniques generate a *single* hierarchy for browsing the database. Dakka et al. [18] introduced a *supervised* approach for extracting useful facets from a collection of text or text-annotated data. The technique in [18] relies on WordNet [16] hypernyms[1] and on a Support Vector Machine (SVM) classifier to assign new keywords to facets. For example, the words "*cat*" and "*dog*" are classified under the "*Animals*" facet, while the words "*mountain*" and "*fields*" go under the "*Topographic Features*" facet. Unfortunately, the supervised learning approach in [18] has limitations. First, the facets that could be identified are, by definition, limited to the facets that appear in the training set. Second, since the algorithm relies on WordNet hypernyms, it is difficult to work on objects annotated with named entities (or even noun phrases), since WordNet has rather poor coverage of named entities. Finally, although the technique in [18] generates high-quality faceted hierarchies from collections of keyword-annotated objects, the quality of the hierarchies built on top of text documents, such as the articles in The New York Times archive, is comparatively low, due to the inability to identify the terms in these documents that should be used for facet construction.

Next, we describe our approach for overcoming these problems.

[1]Hypernym is a word whose meaning includes the meanings of other words, as the meaning of vehicle includes the meaning of car, truck, motorcycle, and so on.

TABLE I

Facets identified by human annotators in a small collection of 1,000 news articles from The New York Times.

Facets
Location
Institutes
History
People
↪Leaders
Social Phenomenon
Markets
↪Corporations
Nature
Event

```
Input: original database $D$, term extractors $E_1, \dots E_k$
Output: annotated database $I(D)$
foreach document $d$ in $D$ do
    Extract all terms from $d$
    /* Compute term frequencies */
    foreach term $t$ in $d$ do
        $Freq_O(t) = Freq_O(t) + 1$
    end
    /* Identify important terms */
    $I(d) = \emptyset$
    foreach term extractors $E_i$ do
        Use the extractor $E_i$ to identify the important terms
        $E_i(d)$ in document $d$
        Add $E_i(d)$ to $I(d)$
    end
end
```

Fig. 1. Identifying important terms within each document

III. Extracting Facets: A Pilot User Study

Before trying to build any algorithmic solution for generating automatically faceted interfaces, we wanted to examine what the biggest hurdle is to generating such interfaces. For this, we decided to run a small pilot study, examining what navigational structures would be useful for people who are browsing a database of news articles. We experimented with the Newsblaster system [19] from Columbia University, which has a news archive with articles from 24 English news sources, dating back to 2001. As part of our efforts to allow easier access to the archive, we examined how to build a faceted interface on top of the archive, which will automatically adapts to the contents of the underlying news collection (or to the query results, for queries that return thousands of documents).

For our initial pilot study, we recruited 12 students majoring either in journalism or in art history. We randomly chose a thousand stories from *The New York Times* archive, and we asked annotators to manually assign each story to several facets that they considered appropriate and useful for browsing. The most common facets identified by the annotators were "*Location*," "*Institutes*," "*History*," "*People*," "*Social Phenomenon*," "*Markets*," "*Nature*," and "*Event*." For these facets, the annotators also identified other "sub-facets" such as "*Leaders*" under "*People*" and "*Corporations*" under "*Markets*."

From the results of the pilot study, we observed one clear phenomenon: the terms for the useful facets do not usually appear in the news stories. (In our study, this phenomenon appeared for 65% of the user-identified facet terms.) Typically, journalists do not use general terms, such as those used to describe facets, in their stories. For example, a journalist writing a story about *Jacques Chirac* will not necessarily use the term "*Political Leader*" or the term "*Europe*" or even "*France*." Such (missing) *context terms* are tremendously useful for identifying the appropriate facets for the story.

After conducting this pilot experiment, it became clear that a tool for the automatic discovery of useful facet terms should exploit some external resource that could return the appropriate facet terms. Such an external resource should provide the appropriate context for each of the terms that we extract from the database. As a result, a key step of our approach is an expansion procedure, in which the "*important terms*" from each news story are expanded with "*context terms*" derived from external resources. The expanded documents then contain many of the terms that can be used as facets. Next, we describe our algorithm in detail, showing how to identify these "*important*" and "*context*" terms.

IV. Automatic Facet Discovery

The results of our pilot study from Section III indicate that general facet terms rarely occur in news articles. To annotate a given story with a set of facets, we normally skim through the story to identify important terms and associate these terms with other more general terms, based on our accumulated knowledge. For example, if we conclude that the phrase "Steve Jobs" is an important aspect of a news story, we can associate this story with general terms such as "personal computer," "entertainment industry," or "technology leaders." Our techniques operate in a similar way. In particular, our algorithm follows these steps:

1) For each document in the database, identify the *important* terms *within* the document that are useful for characterizing the contents of the document (Section IV-A).
2) For each important term in the original document, query one or more external resources and retrieve the *context* terms that appear in the results. Add the retrieved terms in the original document, in order to create an expanded, "context-aware" document (Section IV-B).
3) Analyze the frequency of the terms, both in the original database and the expanded database and identify the candidate facet terms (Section IV-C).

A. Identifying Important Terms

The first step of our approach (see Figure 1) identifies informative terms[2] in the text of each document. We consider the terms that carry important information about the different aspects of a document to be informative. For example, consider a document d that discusses the actions of *Jacques Chirac* during the *2005 G8 summit*. In this case, the set of important

[2]By *term*, we mean single words and multi-word phrases.

terms $I(d)$ may contain the terms

$$I(d) = \{\textit{Jacques Chirac}, \textit{2005 G8 summit}\}$$

We use the next three techniques to achieve this:

- **Named Entities**: We use a named entities tagger to identify terms that give important clues about the topic of the document. Our choice is reinforced by existing research (e.g., [20, 21]) that shows that the use of named entities increases the quality of clustering and improves news event detection. We build on these ideas and use the named entities extracted from each news story as important terms that capture the important aspects of the document. In our work, we use the named entity tagger provided by the LingPipe[3] toolkit.
- **Yahoo Terms**: We use the "Yahoo Term Extraction"[4] web service, which takes as input a text document and returns a list of significant words or phrases extracted from the document.[5] We use this service as a second tool for identifying important terms in the document.
- **Wikipedia Terms**: We developed our own tool to identify important aspects of a document based on Wikipedia entities. Our tool is based on the idea that an entity is typically described in its own Wikipedia page. To implement the tool, we downloaded[6] the contents of Wikipedia and built a relational database that contains (among other things) the titles of all the Wikipedia pages. Whenever a term in the document matches a title of a Wikipedia entry, we mark the term as important. If there are multiple candidate titles, we pick the longest title to identify the important term.
  Furthermore, we exploit the link structure of Wikipedia to improve the detection of important terms. First, we exploit the "*redirect*" pages, to improve the coverage of the extractor. For example, the entries "Hillary Clinton," "Hillary R. Clinton," "Clinton, Hillary Rodham," "Hillary Diane Rodham Clinton," and others redirect to the page with title "Hillary Rodham Clinton." By exploiting the redirect pages, we can capture multiple variations of the same term, even if the term does not appear in the document in the same format as in the Wikipedia page title. (We will also use this characteristic in Step 2, to derive context terms.) In a similar manner, we also exploit the *anchor text* from other Wikipedia entries to find different descriptions of the same concept. Even though the anchor text has been used extensively in the web context [22], we observed that the anchor text works even better within Wikipedia, where each page has a specific topic.

Beyond the three techniques described above, we can also follow alternative approaches in order to identify important terms. For instance, we can use domain-specific vocabularies and ontologies (e.g., from the *Taxonomy Warehouse*[7] by Dow Jones) to identify important terms for a domain. In our current work, due to the lack of appropriate text databases that could benefit from such resources, we do not consider this alternative. Still, we believe that utilizing domain-specific resources for indentifying important terms can be very useful in practice.

The next step of the algorithm uses important document terms to identify additional context terms, relevant to the document.

```
Input: annotated database $I(D)$, ext. resources $R_1, \dots R_m$
Output: contextualized database $C(D)$
$C(D) = \emptyset$
foreach document $d$ in $D$ do
    /* Identify context terms $C(d)$ for $d$ */
    $C(d) = \emptyset$
    foreach important term $t$ in $d$ do
        foreach external resource $R_i$ do
            Query resource $R_i$ using term $t$
            Retrieve context terms $R_i(t)$
            Add $R_i(t)$ to $C(d)$
        end
    end
    Augment $d$ with context terms $C(d)$
end
```

Fig. 2. Deriving context terms using external resources

B. Deriving Context Using External Resources

In Step 2 of our approach, we use the identified important terms to expand each document with relevant context (see Figure 2). As we discussed in Section III, in order to build facets for browsing a text database, we need more terms than the ones that appear in the database. To discover the additional terms, we use a set of *external resources* that can provide the additional context terms when queried appropriately.

For example, assume that we use Wikipedia as the external resource, trying to extract context terms for a document d with a set of important terms $I(d) = \{\textit{Jacques Chirac}, \textit{2005 G8 summit}\}$. We query Wikipedia with the two terms in $I(d)$, and we analyze the returned results. From the documents returned by Wikipedia, we identify additional context terms for the two terms in the original $I(d)$: the term *President of France* for the original term *Jacques Chirac* and the terms *Africa debt cancellation* and *global warming* for the original term *2005 G8 summit*. Therefore, the set $C(d)$ contains the three additional context terms, *president of France*, *Africa debt cancellation*, and *global warming*.

In our work, we use four external resources, and our framework can be naturally expanded to use more resources, if necessary. As part of our research, we used two existing applications (WordNet and Google) that have proved useful in the past and developed two new resources (Wikipedia Graph and Wikipedia Synonyms). In particular, the resources that we use are the following:

[3] http://www.alias-i.com/lingpipe/
[4] http://developer.yahoo.com/
[5] We have observed empirically that the quality of the returned terms is high. Unfortunately, we could not locate any documentation about the internal mechanisms of the web service.
[6] http://en.wikipedia.org/wiki/Wikipedia:Database_download
[7] http://www.taxonomywarehouse.com/

- **Google**: The Web can be used to identify terms that tend to cooccur frequently. Therefore, as one of the expansion strategies, we query Google with a given term, and then retrieve as context terms the most frequent words and phrases that appear in the returned snippets.
- **WordNet Hypernyms**: Previous studies in the area of automatic generation of facet hierarchies [18, 23] observed that WordNet [16] *hypernyms* are good terms for building facet hierarchies. Based on our previous experience [18], hypernyms are useful and high-precision terms, but tend to have low recall, especially when dealing with named entities (e.g., names of politicians) and noun phrases (e.g., "due diligence"). Therefore, WordNet should not be the only resource used but should be complemented with additional resources. We discuss such resources next.
- **Wikipedia Graph**: A very useful resource for discovering context terms is Wikipedia. In particular, the links that appear in the page of each Wikipedia entry can offer valuable clues about associations with other entries. To measure the level of association between the two Wikipedia entries t_1 and t_2 that are connected with a link $t_1 \rightarrow t_2$, we examine two values: the number of outgoing links $out(t_1)$ from t_1 to other entries and the number of incoming links $in(t_2)$ pointing to t_2 from other entries. Using $tf.idf$-style scoring, we set the level of association to $\log(N/in(t_2))/out(t_1)$, where N is the total number of Wikipedia entries. (Notice that the association metric is not symmetric.) When querying the "Wikipedia Graph" resource with a term t, the resource returns the top-k terms[8] with the highest scoring terms. For example, there is a page dedicated to the Japanese samurai "Hasekura Tsunenaga." The "Hasekura Tsunenaga" page is linked to the pages "Japanese Language," "'Japanese," "Samurai," "Japan," and to several other pages. There are more than 6 million entries and 35 million links in the Wikipedia graph, creating an informative graph for deriving context. As expected, the derived context terms will be both more general and more specific terms. We will examine in Section IV-C how we identify the more general terms, using statistical analysis of the term frequencies in the original database and in the contextualized database.
- **Wikipedia Synonyms**: We constructed Wikipedia Synonyms as a resource that returns variations of the same term. As we described in Section IV-A, we can use the Wikipedia redirect pages to identify variations of the same term. To achieve this, we first group together the titles of entries that redirect to a particular Wikipedia entry. For example, the entries "Hillary Clinton," "Hillary R. Clinton," "Clinton, Hillary Rodham," and "Hillary Rodham Clinton" are considered synonyms since they all redirect to "Hillary Rodham Clinton."

  Although redirect pages return synonyms with high accuracy, there are still variations of a name that cannot be captured like this. For such cases, we use the anchor text that is being used in other Wikipedia pages to link to a particular entry. For example, there is a page dedicated to the Japanese samurai "Hasekura Tsunenaga." The "Hasekura Tsunenaga" has also pointers that use the anchor text "Samurai Tsunenaga," which can also be used as a synonym. Since anchor text is inherently noiser than redirects, we use a form of $tf.idf$ scoring to rank the anchor text phrases. Specifically, the score for the anchor text p pointing to a Wikipedia entry t is $s(p,t) = tf(p,t)/f(p)$, where $tf(p,t)$ is the number of times that the anchor phrase p is used to point to the Wikipedia entry t , and $f(p)$ is the number of different Wikipedia entries pointed by the same text p.

[8]We set $k = 50$ in our work.

```
Input: original database $D$, contextualized database $C(D)$
Output: useful facet terms $Facet(D)$
foreach term $t$ in $D$, $C(D)$ do
    $df(t) = 0$, $df_C(t) = 0$
end
foreach document $d$ in $D$, $C(D)$ do
    /* Compute term frequencies in $D$ */
    Extract all terms from $d$
    foreach term $t$ in $d$ do
        $df(t) = df(t) + 1$, $df_C(t) = df_C(t) + 1$
    end
    /* Compute term frequencies in $C(D)$ */
    Extract context terms from $C(d)$
    foreach term $t$ in $C(d)$ do
        $df_C(t) = df_C(t) + 1$
    end
end
foreach term $t$ in $D$, $C(D)$ do
    Compute shift in rank and frequency
    if $Shift_f(t) > 0$ AND $Shift_r(t) > 0$ then
        Add $t$ to facet terms $Facet(D)$
        Compute the Log-Likelihood Statistic $-\log \lambda_t$
    end
end
return the top-$k$ terms in $Facet(D)$, ranked by $-\log \lambda_t$
```

Fig. 3. Identifying important facet terms by comparing the term distributions in the original and in the contextualized database

At the end of Step 2, we create a "*contextualized*" database in which each document contains the original terms *and* a set of context terms. Next, we describe how we can use the term frequencies in the original and in the contextualized database to identify useful facet terms.

C. Comparative Term Frequency Analysis

We now shift our discussion to Step 3 of our approach (Figure 3). So far, we have identified important terms in each document and used them to expand the document with general relevant context for each document. In this step, we process both the expanded and original collections to identify terms that are good candidates for facet terms.

Our algorithm is based on the intuition that facet terms are infrequent in the original database but frequent in the expanded one. So, to identify such terms, we need first to identify terms that occur "more frequently" and then make sure that this difference in frequency is statistically significant, and not simply the result of noise. To measure the difference in frequency, we define the next two functions:

- **Frequency-based Shifting**: For each term t, we compute the frequency difference as
$$Shift_f(t) = df_C(t) - df(t)$$
where $df_C(t)$ and $df(t)$ are the frequencies of term t in the contextualized database and the original database, respectively. Due to the Zipfian nature of the term frequency distribution, this function tends to favor terms that already have high frequencies in the original database. Terms with high frequencies demonstrate higher increases in frequency, even if they are less popular in the expanded database compared to the original one. The inverse problem appears if we use ratios instead of differences. To avoid the shortcomings of this approach, we introduce a rank-based metric that measures the differences in the ranking of the terms.
- **Rank-based Shifting**: We assume that we have a function B that assigns terms to bins based on their ranking. In this paper, we use the function
$$B(t) = \lceil \log_2(Rank(t)) \rceil$$
where $Rank(t)$ is the rank of the term t in the database. After computing the bin $B_D(t)$ and $B_C(t)$ of each term t in the original database and contextualized database, respectively, we define the shifting function to be
$$Shift_r(t) = B_D(t) - B_C(t)$$

In our approach, a term becomes a candidate facet term only if both $Shift_f(t)$ and $Shift_r(t)$ are positive. After identifying terms that occur more frequently in the contextualized database, the next test verifies that the difference in frequency is statistically significant. A test such as the chi-square test could be potentially used to identify important frequency differences. However, due to the power-law distribution of the term frequencies [24], many of the underlying assumptions for the chi-square test do not hold for text frequency analysis [25]. Therefore, we use the *log-likelihood* statistic, assuming that the frequency of each term in the (original and contextualized) databases is generated by a binomial distribution:

- **Log-Likelihood Statistic:** For a term t with document frequency df in the original database D, and frequency df_C in the contextualized database $C(D)$, the log-likelihood statistic for the binomial case is:
$$\begin{aligned} -\log\lambda_t &= \log L(p_1, df_C, |D|) + \log L(p_2, df, |D|) \\ &\quad - \log L(p, df, |D|) - \log L(p, df_C, |D|) \end{aligned}$$
where $\log L(p,k,n) = k\log(p) + (n-k)\log(1-p)$, and $p_1 = \frac{df_C}{|D|}$, $p_2 = \frac{df}{|D|}$, and $p = \frac{p_1+p_2}{2}$. For an excellent description of the log-likelihood statistic see the seminal paper by Dunning [25].

The shift functions and the log-likelihood test return a set of terms $Facet(D)$ that can be used for faceted navigation. Once we have identified these terms, it is then relatively easy to build the actual hierarchies. For our work, we used the subsumption algorithm by Sanderson and Croft [10] that gave satisfactory results, although newer algorithms [5] may give even better results.

V. Experimental Evaluation

In this section, we discuss the experimental evaluation of our techniques. Section V-A discusses the data sets that we used for our evaluation. Then, Sections V-B and V-C describe how we evaluated the recall and precision of our techniques, respectively. Then, Section V-D presents our results on the efficiency of our techniques, and Section V-E briefly discusses some results of a user study that demonstrate the usefulness of the derived faceted navigational structures. Finally, Section V-F provides some further discussion.

A. Data Sets

For our experiments, we used the following data sets:

- **Single Day of NYT (SNYT)**: A collection of 1,000 news stories from The New York Times archive, from one day in November 2005.
- **Single Day of Newsblaster (SNB)**: A collection of 17,000 news stories from one day in November 2005, retrieved by Newsblaster [19] from 24 news sources. We use this data set to test how our techniques work when data come from multiple sources.
- **Month of NYT (MNYT)**: A collection of 30,000 news stories from The New York Times archive, covering one month of news.

B. Recall

Our technique takes as input a set of free-text documents and as outputs a set of hierarchically structured facets that can be used to browse the text database in an OLAP-like slice-and-dice manner. Since there is no standard benchmark for evaluating the quality of the generated facets, we conducted a human study, trying to evaluate the "precision" and "recall" of the generated facets.

Since our experiments required extensive input from users, we launched a large-scale user study using the Amazon Mechanical Turk service.[9] This service offers access to a community of human subjects and tools to distribute small tasks that require human intelligence. In our recall study, each Mechanical Turk annotator had to read a story and identify, for each story, terms that can be used for faceted navigation. We indicated to the annotators that the terms may or may not appear in the document, and it was up to the annotator to determine the terms that were useful for the task of faceted navigation. For each article, the subjects were asked to provide up to 10 candidate facet terms. We instructed the subjects to choose terms that were clearly defined, mutually exclusive, and covered as many aspects, properties, or characteristics of the story as possible.

[9]http://www.mturk.com

politics, money, market, government, history, competition, people, education, location, new york, power, terrorism, war, baseball, event, biography, business, children, development, health, music, real estate, sports, change, comeback, crime, entertainment, greed, national security, nature poverty, spending, success, pride, technology, winning, anger, architecture, branding, foreign lands, bush administration, campaign, capitalism, challenges, civil unrest, civil war, community, compromise, computers, consumer confidence, corruption, countries, culture of fear, disagreement, distribution, power of the Internet, expectations fear, humor, innovation, investigation, Iraq, Italian culture, jobs, leadership, moving, opportunity, optimism, planning, players, police, public relations, publicity, religion, religious, warfare rights, statistics, support, time, torture, U.S., violence, wealth, youth

Fig. 4. A sample of the most frequent facet terms, as extracted by human annotators. All the terms above were anonymously selected by at least two annotators.

year, new, time, people, state, work, school, home, mr, report, game, million, week, percent, help, right, plan, house, high, world, american, month, live, call, thing

Fig. 5. Facet terms identified by a simple subsumption-based algorithm [10], without using our techniques.

Each of the 1,000 stories in *SNYT* was examined by five annotators while for *SNB* and *MNYT* we picked a random sample of 1,000 stories, and again each story was annotated by five annotators. To eliminate annotation errors or idiosyncratic annotations, we considered an annotation to be valid if at least two of the annotators used the same facet term to annotate the story. The final set contained 633 facet terms for *SNYT*, 756 facet terms for *SNB*, and 703 terms for *MNYT*. These numbers indicate that the number of facet terms increases relatively slowly with increased number of news sources (i.e., from *SNYT* to *SNB*) and when expanding the examined time period (i.e., from *SNYT* to *MNYT*). To make sure that this is not an artifact of our sampling approach, we also performed a sensitivity test, examining how the number of facet terms increases as we increase the number of stories in each data set from 100 to 1,000. At 100 documents, we discovered approximately 40% of the facet terms, and at 500 documents we discovered approximately 80% of the facet terms for each of the data sets. Therefore, we believe that the marginal increase in facet terms if we annotated all the 17,000 articles for *SNB* and all the 30,000 articles for *MNYT* would be relatively small. Figure 4 shows a sample of the most frequently identified facet terms for the three data sets.

The next step was to measure how many of the manually extracted facet terms were also identified by our techniques. We define recall as the fraction of the manually extracted facet terms that were also extracted by our techniques. To examine the individual effect of each term extractor (Section IV-A) and of each external resource (Section IV-B), we computed the fraction of identified facet terms for each of the possible combinations of term extractor and external resource. We also computed the recall for the case in which we used *all* the term

TABLE II
THE RECALL OF THE EXTRACTED FACETS, AS JUDGED BY THE HUMAN ANNOTATORS FOR THE *SNYT* DATA SET.

External Resource	**Term Extractors** NE	Yahoo	Wikipedia	*All*
Google	0.529	0.703	0.761	*0.819*
WordNet Hypernyms	0.090	0.510	0.491	*0.592*
Wikipedia Synonyms	0.105	0.156	0.345	*0.408*
Wikipedia Graph	0.632	0.791	0.801	*0.881*
All	*0.746*	*0.891*	*0.899*	**0.945**

TABLE III
THE RECALL OF THE EXTRACTED FACETS, AS JUDGED BY THE HUMAN ANNOTATORS FOR THE *SNB* DATA SET.

External Resource	**Term Extractors** NE	Yahoo	Wikipedia	*All*
Google	0.515	0.658	0.699	*0.751*
WordNet Hypernyms	0.084	0.487	0.395	*0.514*
Wikipedia Synonyms	0.112	0.162	0.306	*0.314*
Wikipedia Graph	0.615	0.755	0.773	*0.823*
All	*0.707*	*0.861*	*0.856*	**0.881**

extractors and *all* the external resources.

We list the results in Tables II, III, and IV for the *SNYT*, *SNB*, and *MNYT* data sets, respectively. The results were consistent across data sets. In general, recall increases as we increase the number of term extractors and as we increase the number of external resources. Wikipedia Synonyms and WordNet Hypernyms tend to perform relatively poorly compared to Google and Wikipedia Graph, especially when using Named Entities as the term extractor. However, both resources are helpful when combined with Google and Wikipedia Graph, and increase the overall recall of the results.

C. Precision

We could also evaluate the precision of the extracted facets using the same methodology that we used for estimating recall. However, our techniques extract a significant number of concepts that the Mechanical Turk annotators did not identify when marking the important facet terms in the documents. Still, when the annotators looked at the generated facet hierarchies, they could easily say whether a particular facet term accurately depicts the contents of the underlying database. So, to estimate precision, we asked the annotators to examine the extracted facet hierarchies and examine the following: (a) whether the facet terms in the hierarchies are useful and (b) whether the term is accurately placed in the hierarchy. We consider a facet term to be "precise" if both conditions are satisfied; the precision is then the ratio of precise terms over the total number of extracted facet terms.

To ensure the quality of the precision annotation, the Mechanical Turk annotators that participated in this experiment had to pass a *qualification test*. To generate our qualification test, we did the following: Initially, we picked random subtrees

TABLE IV

THE RECALL OF THE EXTRACTED FACETS, AS JUDGED BY THE HUMAN ANNOTATORS FOR THE *MNYT* DATA SET.

External Resource	**Term Extractors** NE	Yahoo	Wikipedia	*All*
Google	0.522	0.658	0.699	*0.793*
WordNet Hypernyms	0.087	0.487	0.395	*0.555*
Wikipedia Synonyms	0.109	0.146	0.331	*0.392*
Wikipedia Graph	0.627	0.778	0.790	*0.853*
All	*0.733*	*0.859*	*0.860*	**0.921**

TABLE V

THE PRECISION OF THE EXTRACTED FACETS, AS JUDGED BY THE HUMAN ANNOTATORS FOR THE *SNYT* DATA SET.

External Resource	**Term Extractors** NE	Yahoo	Wikipedia	*All*
Google	0.615	*0.769*	0.751	0.678
WordNet Hypernyms	*0.923*	*0.901*	**0.932**	*0.915*
Wikipedia Synonyms	0.734	0.815	*0.845*	0.819
Wikipedia Graph	0.828	0.813	*0.842*	0.827
All	0.817	0.796	0.858	*0.866*

TABLE VI

THE PRECISION OF THE EXTRACTED FACETS, AS JUDGED BY THE HUMAN ANNOTATORS FOR THE *SNB* DATA SET.

External Resource	**Term Extractors** NE	Yahoo	Wikipedia	*All*
Google	0.505	*0.796*	0.751	0.714
WordNet Hypernyms	*0.897*	*0.919*	*0.909*	**0.922**
Wikipedia Synonyms	0.633	*0.904*	0.875	0.853
Wikipedia Graph	0.789	0.851	*0.885*	0.822
All	0.796	0.815	*0.834*	0.831

TABLE VII

THE PRECISION OF THE EXTRACTED FACETS, AS JUDGED BY THE HUMAN ANNOTATORS FOR THE *MNYT* DATA SET.

External Resource	**Term Extractors** NE	Yahoo	Wikipedia	*All*
Google	0.487	0.818	*0.834*	0.794
WordNet Hypernyms	*0.878*	*0.925*	**0.932**	*0.917*
Wikipedia Synonyms	0.691	0.851	*0.880*	0.879
Wikipedia Graph	0.801	0.824	*0.837*	0.810
All	0.713	0.836	0.855	*0.861*

of the Open Directory[10] hierarchy as our "correct" hierarchies. To generate "noisy" hierarchies, we randomly perturbed some parent-children relations and randomly swapped terms across separate hierarchy subtrees. Then, during the qualification test, each prospective Mechanical Turk annotator had to annotate 20 correct and incorrect hierarchies and was only allowed to proceed with the real annotation task if he or she gave the correct answer for at least 18 out of 20 hierarchies. As in the case of recall measurements, each facet term was examined by five Mechanical Turk annotators. We only consider a facet term as precise if at least four annotators marked the facet term as precise.

We list the precision results in Tables V, VI, and VII for the *SNYT*, *SNB*, and *MNYT* data sets, respectively. Again, the results were consistent across data sets. The highest precision hierarchies are those generated by WordNet; this is not surprising since the hypernyms naturally form a hierarchy. The use of Google as external resource tends to drop precision. In our implementation, for efficiency, we only use the terms that appear in the titles and in the snippets in the Google results; we do not retrieve the actual HTML pages of the returned results. This approach introduces a relatively large number of noisy terms. An interesting direction for future research is to examine whether the introduction of a term extraction mechanism from the HTML pages could improve the precision of our hierachies. In contrast to Google, the use of the Wikipedia resources gives more precise hierarchies. Given the high precision of the WordNet- and Wikipedia-based hierarchies, it would be interesting to see if we could use as external resources ontologies that combine WordNet and Wikipedia in a single resource [26].

[10]http://www.dmoz.org

D. Efficiency

In our experiments, the term extraction took 2-3 seconds per document, and the main bottleneck was the Yahoo! Term Extractor. If we eliminate the Yahoo! Term Extractor, then we can process approximately 100 documents per second. Similarly, the document expansion takes approximately 1 second per document when using Google as an external resource. Using Wikipedia and WordNet, which are stored locally, is significantly faster: we can process more than 100 documents per second, effectively making the term extraction the real bottleneck in the process. The facet term selection phase is extremely fast (a few milliseconds), and we speed up the hierarchy construction using the techniques described in [18], to create the facet hierarchies in 1-2 seconds.

In a real deployment scenario, we can speed up the facet extraction considerably by performing the term and context extraction offline. In this case, the results are ready before the real facet computation, which then takes only a few seconds and is almost independent of the collection size. (So, we can generate facet hierarchies over the complete database *and* dynamically over a set of lengthy query results.) If term and context extraction need to be performed on-the-fly over thousands of documents, and it is important to compute the facet hierarchies fast, then it would be preferable to avoid using web-based resources (Yahoo! Term Extractor and Google), and use only locally available resources (Named Entity Recognizer, Wikipedia, and WordNet) instead.

E. User Study

Finally, we wanted to examine the reaction of users to automatically extracted facets. For this, we recruited five users to use our system to locate news items of interest, and we asked them to repeat the task 5 times. We provided a keyword-based search interface that was augmented with our facet-hierarchies located on the side. We measured how often during each search session the users clicked on the facet terms and how often they used the keyword-based search. We also measured the time required to finish the task, and we asked users to indicate their level of satisfaction at the end on a 0-3 scale (0-dissatisfied, 1-slightly dissatisfied, 2-slightly satisfied, 3-satisfied).

We observed an interesting phenomenon. In the *first* interaction with the system, the users typed as a keyword query a named entity associated with the general topic that they were interested in (e.g., “war in Iraq”). Then they proceeded to locate the news stories of interest by clicking on the facet hierarchies, until they had generated a small subset of news stories associated with the same topic. Interestingly enough, though, in the subsequent interactions with the system, the users started using the facet hierarchies directly and their use of the keyword search interface was gradually reduced by up to 50%. Similarly, the time required to complete each task dropped by 25%, and the level of satisfaction remained statistically steady, with a mean level of 2.5 in the 0-3 scale. These results are consistent with previous research studies that relied on manually generated facet hierarchies [3] or on hierarchies extracted only from WordNet [23].

F. Overall

Overall, the results of our user study indicate that users like facet terms, and use them actively, especially after getting used to their presence. Furthermore, by using the facets, users can locate items of interest faster, without any drop in satisfaction level. The similarity of the interface with existing OLAP tools means that our tools can be seamlessly integrated with current OLAP systems that provide easy access to structured, relational data. Our tools can expand existing OLAP systems to work over unstructured, or semi-structured data, allowing the OLAP users to quickly discover interesting associations (e.g., “show profit-margin distribution for users with this type of complaints”).

VI. Related Work

Faceted interfaces, which use multiple, orthogonal classification schemes to present the contents of a database, have become increasingly popular. A large number of e-commerce web sites use faceted interfaces [27], based on engines provided by companies such as Endeca[11] and Mercado,[12] which expose the facets that have already been defined for the products (e.g., “by price,” “by genre,” and so on). Systems developed in academia, such as HiBrowse [2], OVDL [28], and Flamenco [3], demonstrate the superiority of faceted interfaces over single hierarchies. Our work on the automatic construction of multifaceted interfaces contributes to this area and facilitates the deployment of faceted databases. In an orthogonal direction, Ross and Janevski [29] presented work on *searching* faceted databases and described an associated entity algebra and a query engine.

As an alternative to creating a separate hierarchy for each collection, Chaffee and Gauch [30] presented a system that uses a personalized ontology to offer a common browsing experience across collections of web pages (i.e., web sites) that organize their contents in different ways. Other, less common browsing structures have been proposed (e.g., wavelet-based text visualization [31], dynamic document linking [32]), but hierarchy-based approaches continue to be the most popular interfaces for faceted browsing.

Our approach to identifying facet terms is conceptually similar to the *skew divergence* of Lee [33], which is used to identify substitute terms (e.g., that “fruit” can approximate “apple” but not vice versa). Recent work by Sahami and Heilman [34] identifies semantically similar text snippets (e.g., “UN Secretary-General” and “Kofi Anan”), and could also be useful in our scenario where we are trying to identify generic facet terms that subsume the important terms that appear in our documents. On a broader context, our work relies on *distributional analysis* [33] of two collections (the original and the expanded one) to identify terms that have high distributional differences across the two collections, hoping that these terms are good facets terms. Distributional analysis has also been used by Gabrilovich et al. [35] for novelty detection in a stream of news, and by Cronen-Townsend et al. [36] to measure the “clarity” of a query with respect to a given document collection.

This paper substantially expands our previous work in [37]. Our earlier paper presented preliminary results on extracting useful facet terms by expanding a database using WordNet and then comparing the two collections using the $Shift_f$ and $Shift_r$ metrics from Section IV-C. In the current paper, we show how to use Wikipedia for identifying important terms and for context extraction, and we use the log-likelihood statistic as an additional, more principled mechanism for extracting useful facet terms. Furthermore, we performed extensive user studies in order to evaluate the effectiveness of our techniques.

VII. Discussion and Conclusions

We presented a set of techniques for automatically identifying terms that are useful for building faceted hierarchies. Our techniques build on the idea that external resources, when queried with the appropriate terms, provide useful context that is valuable for locating the facets that appear in a database of text documents. We demonstrated the usefulness of Wikipedia, WordNet, and Google as external resources. Our experimental results, validated by an extensive study using human subjects, indicate that our techniques generate facets of high quality that can improve the browsing experience for users.

We believe that it is relatively straightforward to integrate in this framework other resources that are useful within specialized contexts. For instance, the *Taxonomy Warehouse*[13] by Dow Jones contains a large list of controlled vocabularies and specialized taxonomies that can be used for term identification and term expansion, respectively. For example, when browsing literature for financial topics, we can use one of the available glossaries to identify financial terms in the documents; then, we can expand the identified terms using one (or more) of the

[11]http://www.endeca.com
[12]http://www.mercado.com

[13]http://www.taxonomywarehouse.com/

available financial ontologies and thesauri. If efficiency is not a major concern, we can incorporate multiple such resources in our framework, for a variety of topics, and use all of them, irrespectively of the topics that appear in the underlying collection. The distributional analysis step of our technique (Section IV-C) automatically identifies which concepts are important for the underlying database and generates the appropriate facet terms. We plan to perform more experiments in this direction and examine the performance of our techniques for a larger variety of text databases and external resources.

References

[1] A. G. Taylor, *Wynar's Introduction to Cataloging and Classification*, 10th ed. Libraries Unlimited, Inc, 2006.

[2] A. S. Pollitt, "The key role of classification and indexing in view-based searching," in *Proceedings of the 63rd International Federation of Library Associations and Institutions General Conference (IFLA'97)*, 1997.

[3] K.-P. Yee, K. Swearingen, K. Li, and M. A. Hearst, "Faceted metadata for image search and browsing," in *Proceedings of the 2003 Conference on Human Factors in Computing Systems, CHI 2003*, 2003, pp. 401–408.

[4] P. Wu, Y. Sismanis, and B. Reinwald, "Towards keyword-driven analytical processing," in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data (SIGMOD 2007)*, 2007.

[5] R. Snow, D. Jurafsky, and A. Y. Ng, "Semantic taxonomy induction from heterogenous evidence," in *Proceedings of the 44th Annual Meeting of the Association for Computational Linguistics (ACL 2006)*, 2006.

[6] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J. W. Tukey, "Scatter/Gather: A cluster-based approach to browsing large document collections," in *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'92*, 1992, pp. 318–329.

[7] M. Meila and D. Heckerman, "An experimental comparison of several clustering and initialization methods," *Machine Learning*, vol. 42, no. 1/2, pp. 9–29, 2001.

[8] H.-J. Zeng, Q.-C. He, Z. Chen, W.-Y. Ma, and J. Ma, "Learning to cluster web search results," in *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2004*, 2004, pp. 210–217.

[9] M. A. Hearst and J. O. Pedersen, "Rexamining the cluster hypothesis: Scatter/Gather on retrieval results," in *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'96*, 1996, pp. 76–84.

[10] M. Sanderson and W. B. Croft, "Deriving concept hierarchies from text," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'99*, 1999, pp. 206–213.

[11] D. J. Lawrie and W. B. Croft, "Discovering and comparing hierarchies," in *Recherche d'Information Assistée par Ordinateur (RIAO 2000)*, 2000, pp. 314–330.

[12] C. G. Nevill-Manning, I. H. Witten, and G. W. Paynter, "Lexically-generated subject hierarchies for browsing large collections," *International Journal on Digital Libraries*, vol. 2, no. 2-3, pp. 111–123, 1999.

[13] G. W. Paynter and I. H. Witten, "A combined phrase and thesaurus browser for large document collections," in *Research and Advanced Technology for Digital Libraries, 5th European Conference (ECDL 2001)*, 2001, pp. 25–36.

[14] G. W. Paynter, I. H. Witten, S. J. Cunningham, and G. Buchanan, "Scalable browsing for large collections: A case study," in *Proceedings of the Fifth ACM Conference on Digital Libraries (DL 2000)*, 2000, pp. 215–223.

[15] J. Kominek and R. Kazman, "Accessing multimedia through concept clustering," in *Proceedings of the 1997 Conference on Human Factors in Computing Systems, CHI 1997*, 1997, pp. 19–26.

[16] C. Fellbaum, *WordNet: An Electronic Lexical Database*. MIT Press, May 1998.

[17] E. Stoica and M. A. Hearst, "Nearly-automated metadata hierarchy creation," in *HLT-NAACL 2004: Short Papers*, 2004, pp. 117–120.

[18] W. Dakka, P. G. Ipeirotis, and K. R. Wood, "Automatic construction of multifaceted browsing interfaces," in *Proceedings of the 2005 ACM Conference on Information and Knowledge Management (CIKM 2005)*, 2005, pp. 768–775.

[19] K. McKeown, R. Barzilay, J. Chen, D. K. Elson, D. K. Evans, J. Klavans, A. Nenkova, B. Schiffman, and S. Sigelman, "Columbia's Newsblaster: New features and future directions," in *Proceedings of HLT-NAACL 2003, Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2003.

[20] V. Hatzivassiloglou, L. Gravano, and A. Maganti, "An investigation of linguistic features and clustering algorithms for topical document clustering," in *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2001*, 2001, pp. 224–231.

[21] E. Filatova and V. Hatzivassiloglou, "Marking atomic events in sets of related texts," in *Recent Advances in Natural Language Processing, Part III*, 2003, pp. 247–256.

[22] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *Proceedings of the Seventh International World Wide Web Conference (WWW7)*, 1998, pp. 107–117.

[23] E. Stoica, M. A. Hearst, and M. Richardson, "Automating creation of hierarchical faceted metadata structures," in *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL 2007)*, 2007.

[24] G. K. Zipf, *Human Behavior and the Principle of Least Effort*. Addison-Wesley, 1949.

[25] T. Dunning, "Accurate methods for the statistics of surprise and coincidence," *Computational Linguistics*, vol. 19, no. 1, pp. 61–74, Mar. 1993.

[26] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A core of semantic knowledge," in *Proceedings of the 16th International World Wide Web Conference (WWW 2007)*, 2007, pp. 697–706.

[27] K. La Barre, "Adventures in faceted classification: A brave new world or a world of confusion?" in *8th International Conference of the International Society for Knowledge Organization (ISKO 2004)*, 2004.

[28] G. Marchionini and G. Geisler, "The open video digital library," *D-Lib Magazine*, vol. 8, no. 12, Dec. 2002.

[29] K. A. Ross and A. Janevski, "Querying faceted databases," in *Proceedings of the Second Workshop on Semantic Web and Databases*, 2004.

[30] J. Chaffee and S. Gauch, "Personal ontologies for web navigation," in *Proceedings of the 2000 ACM Conference on Information and Knowledge Management (CIKM 2000)*, 2000, pp. 227–234.

[31] N. E. Miller, P. C. Wong, M. Brewster, and H. Foote, "Topic islands: A wavelet-based text visualization system," in *Proceedings of the conference on Visualization (VIS'98)*, 1998, pp. 189–196.

[32] G. Golovchinsky, "Queries? Links? Is there a difference?" in *Proceedings of the 1997 Conference on Human Factors in Computing Systems, CHI 1997*, 1997, pp. 407–414.

[33] L. Lee, "Measures of distributional similarity," in *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics (ACL 1999)*, 1999.

[34] M. Sahami and T. D. Heilman, "A web-based kernel function for measuring the similarity of short text snippets," in *Proceedings of the 15th International World Wide Web Conference (WWW 2006)*, 2006, pp. 377–386.

[35] E. Gabrilovich, S. Dumais, and E. Horvitz, "Newsjunkie: Providing personalized newsfeeds via analysis of information novelty," in *Proceedings of the 13th International World Wide Web Conference (WWW 2004)*, 2004, pp. 482–490.

[36] S. Cronen-Townsend, Y. Zhou, and W. B. Croft, "Predicting query performance," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2002*, 2006, pp. 299–306.

[37] W. Dakka, R. Dayal, and P. Ipeirotis, "Automatic discovery of useful facet terms," in *ACM SIGIR 2006 Workshop on Faceted Search, 2006*, 2006.