Dynamic Category Sets: An Approach for Faceted Search

Daniel Tunkelang
Endeca
55 Cambridge Parkway
Cambridge, MA 02142
dt@endeca.com

ABSTRACT

In this paper, we present Dynamic Category Sets, a novel approach that addresses the vocabulary problem for faceted data. In their paper on the vocabulary problem, Furnas et al. note that "the keywords that are assigned by indexers are often at odds with those tried by searchers." Faceted search systems exhibit an interesting aspect of this problem: users do not necessarily understand an information space in terms of the same facets as the indexers who designed it. Our approach addresses this problem by employing a data-driven approach to discover sets of values across multiple facets that best match the query. When there are multiple candidates, we offer a clarification dialog that allows the user to disambiguate them.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval – *query formulation, retrieval models, search process, selection process*.

General Terms

Algorithms, Human Factors.

Keywords

Faceted Search, Faceted Navigation.

1. INTRODUCTION

Neither direct search nor navigational search adequately address the information access problem. Direct search against a collection of records appeals to users by offering the simplicity of a text box, but offers no facility for query refinement when searches return unsatisfying results. Navigational search provides guidance through the use of a hierarchical taxonomy, but results in a limited user experience—particularly for information spaces whose records do not have a natural hierarchical organization.

Faceted search aims to combine navigational and direct search to leverage the best of both approaches. In a typical faceted search interface, users start by entering a query into a search box. The system uses this query to perform a full-text search, and then offers navigational refinement on the results of that search. Importantly, faceted search systems assume that the records are organized into multiple independent facets, rather than a single taxonomy. Faceted search has received intense attention in the last few years [4, 6].

In addition to the above search paradigms, there is another one that uses the search box--namely, category search. Category search, which complements either navigational or faceted search approaches, is not a direct search against the records; rather, it searches the space of facet values. For example, the Yahoo directory [7] and the Open Directory Project [8] allow users to search for categories in their taxonomies using a search box.

Sometimes a single search box is used for both direct search and category search. In these cases, systems generally promote category search results, particularly when the assignment of categories to records represents human-entered metadata and thus leads to more precise results than direct search on the record text.

While direct search returns a set of records that can be further refined using a faceted search approach, category search provides results that are themselves entry points into faceted navigation. If a user starts by performing a category search and then selects a result of that search, the resulting navigation state is identical to that achieved if the user selects the same facet value through navigation.

2. THE VOCABULARY PROBLEM

In their paper on the "vocabulary problem" [1], Furnas et al. note that "the keywords that are assigned by indexers are often at odds with those tried by searchers." Faceted search systems exhibit an interesting aspect of this problem: users do not necessarily understand an information space in terms of the same facets as the indexers who designed it. For example, a user searching a collection of books may imagine "20th Century Fiction" as a single value for the Subject facet, while the indexer may have expressed this concept as the intersection of two independent facet values: Time Period = 20th Century and Genre = Fiction. If the user enters "20th Century Fiction" into a category search box, many systems will respond that there are no exact category matches--at best, they will return the facets that partially match the query.

Ideally, an information space would have a unique faceted representation. Unfortunately, such a representation does not usually exist—and, even if it did exist, there is no reason to believe that non-expert users would be familiar with it. Generally, the construction of a faceted information space is a subjective process—there is no right answer to questions like whether European History should be a single value in the Subject facet or

SIGIR '06 Workshop on Faceted Search Conference, August 10, 2006, Seattle, WA, USA.

whether it should be the intersection of Subject = History and Location = Europe.

Indeed, for complex information spaces, there are as many plausible facet models are there are information architects. And, perhaps more importantly, each user has a different preconception of how the space should be arranged. This diversity of models is the vocabulary problem for faceted search.

This vocabulary problem has significant implications for the effectiveness of category search in faceted search systems. In a faceted search system, category search returns individual facet values as results. Hence, if a user enters a query corresponding to multiple facet values, the system cannot possibly return a single result corresponding to the user's intent. Following up on our earlier example, if the user enters "20th Century Fiction" as a query, the system can at best return as options the individual facet values Time Period = 20th Century and Genre = Fiction. While the user may eventually find the desired results, the system is not being particularly helpful in the process.

In the remainder of this paper, we present Dynamic Category Sets as a technique for addressing this vocabulary problem. We note that there are many other aspects of the vocabulary problem, such as synonyms (multiple ways to express the same concept) and polysemy (one word or phrase corresponding to multiple distinct concepts). While Dynamic Category Sets do not address these problems, they can be used in conjunction with linguistic techniques that do.

3. DYNAMIC CATEGORY SETS

Dynamic Category Sets (DCS) generalize category search to overcome the aforementioned vocabulary problem. While category search returns individual facet values as results, Dynamic Category Sets (DCS) return results that are themselves sets of facet values. For example, a search for “20th Century Fiction” might return the two sets {Time Period = 20th Century, Genre = Fiction} and {Time Period = 20th Century, Genre = Non-Fiction} as results. A more elaborate example: a search for “audio history” might return {Media Type = Audio, Subject = History} and {Subject = Audio Technology, Subject = History} as results. We note that, in this last result, a single work is classified using multiple values from the same Subject facet. DCS can be used with both single-assign (i.e., at most one value assigned from each facet) and multi-assign faceted models.

Ordinary category search results remain an important special case for DCS—they appear in the results as singleton sets. For example, a search for “audio technology” might return {Subject = Audio Technology} and {Media Type = Audio, Subject = Technology} as results. When there are singleton results that match the user’s query, they are often the best matches. DCS is primarily intended to address the case where there is no single facet value that adequately represents the user’s intent but we still want to achieve the precision associated with category search.

We interpret the sets returned by DCS as intersections—that is, each set of facet values in the results corresponds to the selection of all its facet values. For example, {Time Period = 20th Century, Genre = Fiction} means works that satisfy both of these criteria. There are other possible interpretations (e.g., using partial match semantics), but intersection provides an intuitive interpretation that aligns well with the faceted search paradigm. In effect, we derive from existing facets a precisely defined, previously non-existent category.

DCS results are required to satisfy three properties: they are data-driven; they match the search query; and they are minimal. We first discuss these three requirements, and then consider the computation, rendering, and organization of DCS results.

3.1 Data-Driven

While there may be many possible sets of facet values that match the query (we will discuss matching semantics shortly), not all sets are useful search results. In particular, we do not want to present results to a user that lead to no results, i.e., a “dead end” in the information space.

Hence, a search for “20th Century Bach” might return {Time Period = 20th Century, Composer = P. D. Q. Bach} and {Time Period = 20th Century, Author = Richard Bach} but not {Time Period = 20th Century, Composer = Johann Sebastian Bach}, since Johann Sebastian Bach died in 1750, and the intersection of Time Period = 20th Century and Composer = Johann Sebastian Bach is empty.

By making DCS results data-driven, we not only ensure that the results are meaningful, but we also dramatically cull the space of possible combinations of facet values. Indeed, the set of combinations of facet values that lead to results in the information space typically represents a tiny fraction of the space of theoretically possible combinations.

Eliminating dead ends is perhaps the most important data-driven means of constraining DCS results. Another option is to impose a minimum threshold for the number of records associated with a result. We will return to this option when we discuss filtering in a later section on organizing DCS results.

3.2 Matching Semantics

The preceding examples of DCS results illustrate that different interpretations of a search query can produce different corresponding sets of facet values. We now look more formally at the notion of matching semantics for DCS.

The simplest (and preferred) semantics are match-all: that is, a DCS result is required to consume all of the query terms. All of our examples so far have followed this model. For example, the set {Time Period = 20th Century, Genre = Fiction} consume the three query terms in the search “20th Century Fiction”.

We can relax match-all to allow for stop words (e.g., the, of) and stemming (e.g., singular/plural forms of nouns). For example, if the query is “histories of France in the 20th century”, we might accept {Time Period = 20th Century, Subject = History, Location = France} even though it omits various stop words (of, in, the) and consumes the term “histories” only by way of stemming.

While match-all semantics generally lead to precise results, it may fail to produce any results at all. For example, there may be no combination of facet values that entirely consume the query “histories of France in the 20th century”.

In order to improve recall at the possible expense of precision, we can use match-partial semantics. In the above example, there may be combinations of facet values that match most of the query terms, e.g., {Time Period = 20th Century, Location = France} or

{Subject = History, Location = France}. While these are not perfect matches, they represent a best effort to match the query.

Match-partial semantics specify how many of the query terms must be consumed and how many can be omitted. For example, we may allow results that match at least half of the query terms and omit at most two terms. There are many possible ways to implement match-partial semantics, each with different tradeoffs for precision and recall.

3.3 Minimality

We have discussed how DCS results are data driven and are required to match the query according to specified matching semantics. The third requirement is that DCS results be minimal.

We motivate minimality with an example. Consider a search query comprised of the single term “fiction”. {Genre = Fiction} and {Genre = Non-Fiction}, the results that an ordinary category search might return, seem reasonable, while results like {Time Period = 20th Century, Genre = Fiction} and {Subject = History, Genre = Fiction} do not. These latter results are confusing because they are more specific than needed to match the query. Following Grice’s maxim of quantity [2], we prefer minimal results that are informative enough but not more informative than necessary.

We define a DCS result as minimal if it is no larger than necessary—that is, no proper subset of the facet values in the result yields a match. Hence {Time Period = 20th Century, Genre = Fiction} is not a minimal match for “fiction”, since removing its first value yields the match {Genre = Fiction}.

If we are using match-partial semantics, we can generalize minimality by considering a result to be minimal if removing a value from the result yields a set that either is not a match or matches fewer query terms.

Minimality is a key characteristic of the DCS approach: without minimality, DCS would degenerate to direct search. In effect, every record that contained a set of facet values matching the query would be returned as a DCS result.

Instead, minimality ensures that DCS returns results that are tightly bound to the query. Indeed, when there are multiple DCS results for a query, they offer query disambiguation or refinement at the highest level possible in the faceted information space.

3.4 Computation

How do we compute DCS results that are data-driven, match the search query, and satisfy minimality?

Let us consider the two brute force approaches.

One approach is to consider all possible combinations of facet values, and then determine, for each combination, whether it leads to results, matches the search query, and satisfies minimality. This top-down approach is only viable for very simple information spaces, as its cost grows at least exponentially in the number of facets. For example, in an information space with ten facets, each having ten values, there are over ten billion (10^{10}) possible combinations of facet values—and as many as 2^{100} possible combinations if we allow for multiple assignment of values from the same facet.

A second approach is to enumerate through all of the records in the information space, consider for each record all of the subsets of its facet values, and then determine which of these match the search query and satisfy minimality. The cost of this approach is the product of the number of records with the number of subsets of facet values per record, the latter quantity being exponential in the number of values. For a collection of one million records, each of which has ten facet values, the total number of subsets considered is $2^{10}*10^{6}$—much better than the previous approach but still prohibitive.

Our approach combines the core elements of the above two approaches, but uses search indexes to make the problem tractable. By leveraging indexes of both the facet values and the records, we dramatically reduce the space of candidate DCS results.

Our first step uses an inverted index mapping words to the facet values containing them, to obtain the set of facet values that at least consume one query term. This step is essentially the same as performing a category search with match-any semantics. Any DCS candidate that satisfies the matching semantics and minimality will be composed entirely of facet values from this set.

Our second step uses an inverted index mapping words to records, to obtain the records whose facet values consume sufficiently many query terms to satisfy the matching semantics. This step is a restricted form of direct search. In order for a DCS candidate to satisfy the data-driven requirement, there must be at least one record in this set of records that contains all of the facet values of that candidate.

We now iterate through the set of records from the second step, looking for sets of facet values from the first step. We have one last optimization: if a set S of facet values does not satisfy minimality, then no superset of S can satisfy minimality. This observation allows us to prune most of the remaining search space.

As a result, we have found it possible to return DCS results at speeds that meet the needs of interactive user applications, i.e., sub-second response in high-volume applications.

3.5 Rendering Search Results

A DCS result is a set of facet values. There are many ways that we can render such sets of values in a user interface.

The simplest is the convention we have adopted in the text: a bracketed sequence of facet-value pairs {Facet1 = Value1, Facet2 = Value2, …}. The convention is transparent, but it is quite verbose, especially for sets of more than two values.

Another approach is to create a breadcrumb similar to the convention for locating a category in a hierarchical taxonomy, e.g., Time Period = 20th Century > Genre = Fiction. This style may raise the objection that it introduces a spurious ordering of the facet values, but users are unlikely to make any inferences from this order.

A variation of this breadcrumb approach is to eliminate the facet names for brevity, e.g., 20th Century > Fiction. This abbreviated approach is particularly appropriate for larger sets of values. It hides the mechanics of DCS entirely: even a sophisticated user is likely to assume that the DCS results are categories residing in a single taxonomy. Hence, there is again the risk of confusing the user, but this time because of erring on the side of brevity.

Nonetheless, in a brief experiment where both novice and expert users were exposed to this style at an online apparel site, all found this rendering style intuitive, as evidenced by the fact that they perceived the DCS results as if they were conventional category search results.

3.6 Organizing Search Results

When there are one or only a few DCS results, we can simply present them all to the user as alternative query interpretations. If the number of results is large, however, we have to consider ways to filter or organize them.

3.6.1 Filtering Search Results

There are at least two techniques we can use to filter results.

The first technique is to limit the number of facet values combined to create a DCS result. For example, we could limit DCS results to combinations of two facet values: thus, for a query of "used science fiction", we would allow {Category = Used, Genre = Science Fiction} as a match but not {Category = Used, Genre = Non-Fiction, Subject = Science}. Filtering based on the number of facet values mitigates the additional complexity introduced by DCS.

The second technique is to filter results based on the number of records associated with a DCS result. All else equal, we favor results associated with more records because they are more likely to contain materials of interest to the user. Also, by culling DCS results that correspond to very small result sets (e.g., a single record), we may avoid spurious combinations of values that result from data errors.

How many facet values are too many? How many records are too few? Here, unfortunately, the answers are mostly a matter of taste. In practice, however, the limited screen real estate of an application may dictate how many DCS results we can show--in which case we may simply want to show the DCS results associated with the most records. We also note that, since we are treating combinations of facet values as intersections, combinations of many facet values will generally be associated with fewer records than combinations of fewer facet values.

3.6.2 Grouping Search Results

Since the DCS results for a query may correspond to various different sets of facets, we may prefer to address this heterogeneity through grouping rather than filtering. A conventional approach for category search on faceted information spaces is to organize the results, which are individual facet values, by facet (e.g., Author, Composer). Since each DCS result is a set of facet values, we extrapolate from this approach, making each group correspond to a set of facets.

For example, a search for “Bach box set” might return {Packaging: Box Set, Composer = Johann Sebastian Bach } and {Packaging: Box Set, Composer = P. D. Q. Bach} in a {Packaging, Composer} group and {Packaging: Box Set, Author = Richard Bach} in a {Packaging, Author} group.

Within each group, we can present results in lexicographic order or in descending order of the number of associated records.

Finally, if the set of DCS results is too large to display, it can be presented through a faceted navigation interface. We do not recommend this approach for general-purpose user interfaces, since it may confuse users with the faceted navigation interface that they use to browse the records themselves.

4. EXAMPLES

4.1 CompUSA

CompUSA is a leading retailer of computers and consumer electronics. Their information space is a particularly good fit for the DCS approach. While their products are well described by facets, many of their users have limited familiarity with the vocabulary of the space. Moreover, the assignment of values to facets is not always clear to novice users or even experts—for example, is “Windows” an “Operating System” or a “Platform”?

Figure 1 illustrates the DCS results of a search for “viewsonic lcd”. All of the results include two facet values, one of which is the unambiguous Brand = ViewSonic. The variation in the second value communicates the multiple meanings of “lcd” in the content and offers a clarification dialogue through the four options:

- Display Type = Active Matrix LCD (TFT)
- Projector Technology = LCD
- Department = Monitors » Flat Panel (LCD)
- Department = Electronics » Televisions » LCD TVs

Presenting each value in the context of its facet allows the user to accurately interpret the sense carried by that instance. In the process, users learn about the information space and may discover the query that expresses their intent more precisely than any that they would have invented on their own.

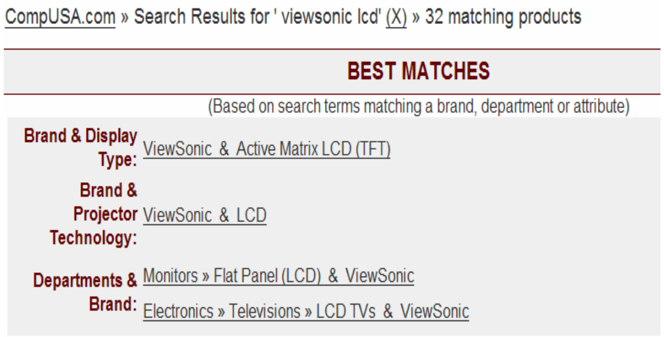
4.2 Enron Email Explorer

In the course of investigating Enron, the United States Federal Energy Regulatory Commission released a collection of about half a million email messages from Enron’s email archives [3]. Thanks to the efforts of Leslie Kaelbling, William Cohen, and their colleagues, this collection has become a valuable resource for researchers in information retrieval and related fields.

Figure 2 illustrates an application that includes DCS as a means of exploring this email application. The example query, “gramm committee” turns out to have various interpretations:

- There are two Gramms: Phil Gramm and Wendy Gramm.
- Phil Gramm was involved with at least two committees, the Banking Committee and the Commerce Committee.

Figure 1: Search for “viewsonic lcd” on CompUSA



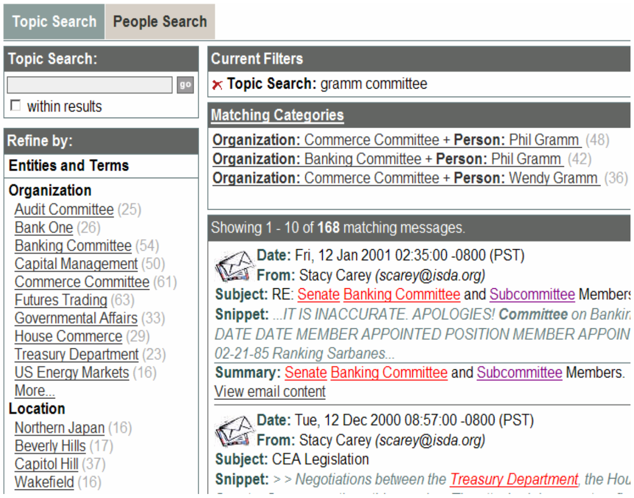
CompUSA.com » Search Results for ' viewsonic lcd' (X) » 32 matching products

BEST MATCHES

(Based on search terms matching a brand, department or attribute)

Brand & Display Type:	ViewSonic & Active Matrix LCD (TFT)
Brand & Projector Technology:	ViewSonic & LCD
Departments & Brand:	Monitors » Flat Panel (LCD) & ViewSonic Electronics » Televisions » LCD TVs & ViewSonic

Figure 2: Search for "gramm committee" on Enron Explorer



Topic Search:

go

□ within results

Refine by:

Entities and Terms

Organization

Audit Committee (25)
Bank One (26)
Banking Committee (54)
Capital Management (50)
Commerce Committee (61)
Futures Trading (63)
Governmental Affairs (33)
House Commerce (29)
Treasury Department (23)
US Energy Markets (16)
More...

Location

Northern Japan (16)
Beverly Hills (17)
Capitol Hill (37)
Wakefield (16)

Current Filters

Topic Search: gramm committee

Matching Categories

Organization: Commerce Committee + **Person:** Phil Gramm (48)
Organization: Banking Committee + **Person:** Phil Gramm (42)
Organization: Commerce Committee + **Person:** Wendy Gramm (36)

Showing 1 - 10 of **168** matching messages.

Date: Fri, 12 Jan 2001 02:35:00 -0800 (PST)
From: Stacy Carey *(scarey@isda.org)*
Subject: RE: Senate Banking Committee and Subcommittee Members
Snippet: *...IT IS INACCURATE. APOLOGIES! **Committee** on Bankir*
DATE DATE MEMBER APPOINTED POSITION MEMBER APPOIN
02-21-85 Ranking Sarbanes...
Summary: Senate Banking Committee and Subcommittee Members.
View email content

Date: Tue, 12 Dec 2000 08:57:00 -0800 (PST)
From: Stacy Carey *(scarey@isda.org)*
Subject: CEA Legislation
Snippet: *> > Negotiations between the Treasury Department, the Hou*

As it turns out, there is a story connecting the Gramms and the committees: a press release from the advocacy organization Public Citizen asserts that Senator Phil Gramm and his wife, Wendy Gramm, at one point chairwoman of the Commodity Futures Trading Commission, acted in a way that reflected an unethical conflict of interest [5].

Here DCS goes beyond query disambiguation to provide insight about the relationships among the facet values it combines. Following up on the DCS results leads to the various emails that substantiate this conflict of interest story.

5. CONCLUSION

DCS helps address the vocabulary problem in faceted search. Specifically, it allows category search to be effective in a faceted information space, even when users do not necessarily model the space using the same facets as the indexer. By employing a data-driven approach to discover sets of values across multiple facets that best match the query, DCS provides a dialogue that helps users to disambiguate queries and to even make new discoveries about the data collection.

DCS complements linguistic and statistical approaches for query interpretation. For example, the matching semantics can leverage familiar query interpretation techniques, such as stop-word elimination, stemming, thesaurus expansion, spelling correction, and automatic phrase detection.

Nonetheless, DCS is quite different from traditional natural language processing (NLP) approaches. DCS takes a combinatorial approach that does not depend on parsing but instead leverages the power of data-driven faceted search. As such, DCS is highly language-independent, and is even effective in non-linguistic contexts (e.g., when facet values contain numbers or arbitrary identifiers). Moreover, since DCS results are sets of facet values rather than parse trees, they do not impose any special requirements (e.g., syntactic analysis) on the indexing of the documents in the information space.

6. ACKNOWLEDGMENTS

My thanks to my colleagues at Endeca for their many contributions to developing the ideas in this paper, and to the anonymous reviewers for their helpful suggestions.

7. REFERENCES

[1] Furnas, G.W., Landauer, T.K., Gomez, L.M., Dumais, S.T.: The vocabulary problem in human-system communication. *Communications of the ACM 30* (1987) 964-971.

[2] Grice, H. P., Logic and conversation, in *Syntax and Semantics, Vol. 3, Speech Acts*, ed. by Peter Cole and Jerry L. Morgan, New York: Academic Press 1975, 41–58.

[3] Klimt, B., & Yang, Y. (2004). Introducing the Enron Corpus, *First Conference on Email and Anti-Spam (CEAS)*, Mountain View, CA, 2004.

[4] Marchionini, G. and Brunk, B., Toward a general relation browser: A GUI for information architects. In *Journal of Digital Information*, volume 4, 2003.

[5] Slocum, T, *Blind Faith: How Deregulation and Enron's Influence Over Government Looted Billions from Americans*, 2001, http://www.citizen.org/documents/Blind_Faith.PDF.

[6] Yee, K-P., Swearingen, K., Li, K., and Hearst, M., Faceted Metadata for Image Search and Browsing. In *CHI 2003*.

[7] http://dir.yahoo.com.

[8] http://dmoz.org/.