Technical Report for SF1 Faceted Search

Jia Guo,Jinli Liu,Yingfeng Zhang

January 25, 2011

Document History		
Date	Author	Content
2011-01-24	Jia Guo	Initial version, describe the background and the baseline approach.
2011-01-24	Jinli Liu	Second version, describe the context-sensive semantic smoothing method used in faceted search.

Abstract

Contents

1 Introduction

Faceted search, also called faceted navigation or faceted browsing, is a technique for accessing a collection of information represented using a faceted classification, allowing users to explore by filtering available information. A faceted classification system allows the assignment of multiple classifications to an object, enabling the classifications to be ordered in multiple ways, rather than in a single, pre-determined, taxonomic order. Each facet typically corresponds to the possible values of a property common to a set of digital objects.

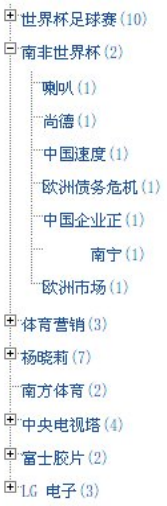
Facets are often derived by analysis of the text of an item using entity extraction techniques or from pre-existing fields in the database such as author, descriptor, language, and format. This approach permits existing web-pages, product descriptions or articles to have this extra metadata extracted and presented as a navigation facet. This type of faceted search could be called as *dynamical facets*. Under this dynamical mode, the representation of facets maybe different depends on the documents set. Take an example of our *taxonomy generation*. There's a result for query "football" in Chinese on one collection, see

Figure 1: TG result of “football“ in Chinese

1.

Also a result for query “football worldcup“ in Chinese on the same collection, see 2.

Figure 2: TG result of “football worldcup“ in Chinese



You can see that the fathership and siblingship is not the same according to the different documents set. So it is dynamic and the facets was generated automatically in runtime. So our “Taxonomy Generation“ is also a kind of *faceted search* and the categories source(concept entities) is extracted from collection by *KPE*.

Another type of “faceted search“ is in a static representation where all categories were predefined in a specific hierarchical form. Like *dmoz*, see 3:

This kind of faceted search is the topic of this report and will play a important role in SF1-R together with TG.

Figure 3: dmoz hierarchical categories

2 Faceted Search in SF1-R

Our faceted search will be built through a user given ontology. Ontology is a basic of categories of being and their relations. There're several ways to describe an ontology in text. One famous and widely used approach is OWL.

2.1 OWL

The Web Ontology Language (OWL) is a family of knowledge representation languages for authoring ontologies. The languages are characterised by formal semantics and RDF/XML-based serializations for the Semantic Web. OWL is endorsed by the World Wide Web Consortium (W3C) and has attracted academic, medical and commercial interest.

We can define the *Operating systems* category in this OWL way:

```
<owl:Class rdf:ID="Operating systems">
<rdfs:subClassOf rdf:resource="#Computer Science"/>
<rdfs:label>Windows XP, Windows NT, Linux kernel, Unix, BSD, OpenVMS</rdfs:label>
</owl:Class>
```

It is very intuitive: this *Operating systems* is a sub class of *Computer Science* and has some labels like "Windows XP" and "Unix". This can be easily defined by users or system managers. An OWL file is consist of a group of classes like above, with the fathership and siblingship relations by *subClassOf* attribute.

2.2 Automatic Categorizing

After finished this OWL file and pass it to SF1-R, users can assign any document in collection to more than one categories defined. However, how about we have millions of documents? It is impossible to do all the assignment manually.

So the most important task of static faceted search is to categorize the documents automatically depends on the meta-data of given ontology. Each document should be categorized into some classes defined in OWL based on their labels, the relationship among all categories and some else information. This seems like a Categorize Problem, but we don't have training data, instead,

we have staic or fixed user-defined classes with given labels(Attention: once user defines these labels, the labels are fixed and can't be revised or tokenized!!! That is why Proximity Language model can't be applied to facted search.), and the goal is to categorize each document to one or more classes. This is similarity compare problem. If we take one class with its given labels as query, and we compute simliarty between this query and document, then the problem can also be seen as an Information Retrieval problem, such that we can use ranking methods to resolve this problem.

language modeling approach is often applied to information retrieval. In this approach, the relevance of a document to a query is defined as the generative probability of the query by the underling model of the document.

$$Rel(Q, d) \propto p(Q|d) \tag{2.1}$$

In the simple case, the query terms are assumed to be independent of each other. The likelihood of the query by the document can be decomposed into

$$p(Q|d) = \prod_i p(q_i|d) \tag{2.2}$$

the model parameter $p(q_i|d)$ can be estimated from each document.

This simple aprroach is Unigram Language Model. Multinomial distributions are often assumed for Information Retrival. With this assumption, the Unigram language model can be simply estimated by a maximum likelihood estimator:

$$p_{ml}(w|d) = \frac{c(w, d)}{\sum_i c(w_i, d)} \tag{2.3}$$

where $c(w, d)$ denotes the count of word w in the document d. This means the probability will be zero if a word never appears in the training document. Such zero probability should be prevented. Otherwise, the product of all word probabilities will be zero, no matter how important other words are.

To prevent zero probability, the raw language model should be smoothed. There are many methods for smoothing, such as Jelinek-Mercer smoothing and Dirichlet smoothing.

The Jelinek-Mercer [1] method linearly interpolates the maximum likelihood model with a corpus model (also referred to as background collection model).

$$p_b(w|d) = (1 - \lambda)P_{ml}(w|d) + \lambda p(w|C) \tag{2.4}$$

where $P_{ml}(w|d)$ and $p(w|C)$ are the maximum likelihood estimator of the document model and corpus model respectively; and the coefficient λ controls the influence of the corpus model in the mixture model.

Dirichlet smoothing[2] assumes that words in the document follow Dirichlet distribution. Each word has a prior count as the parameter of the distribution. Zhai and Lafferty used a corpus model to set the Dirichlet parameters, and the word probability after smoothing becomes

$$p_b(w|d) = \frac{c(w, d) + \mu p(w|C)}{\sum_i c(w_i|d) + \mu} \tag{2.5}$$

The above background language model only makes use of single word, there are many synonyms and polysemy, for example: "mouse"in different context

has different meaning, if there is computer, it means a tool, if there is cat, it is animal. If we don't have any context information, there will be ambiguous and meaningless. If we can make use of context information, it should be more valuable than just single word or n-gram.

Zhou[3] proposed one semantic smoothing method for language model, which incorporates human knowledge or word semantics into the language model estimates, so that it can make full use of context information. The topic signature language model (TSLM) is one of such semantic smoothing methods, it uses multi-word phrase or ontology concept as topic signature, and mapping topic signature to query terms, so that the sematic model is as follows:

$$p_t(w|d) = \sum p(w|t_k)p_{ml}(t_k|d) \tag{2.6}$$

where t_k is the k_th topic signature extracted from document d. The likelihood of a given document generating the topic signature t_k can be estimated with

$$p_{ml}(t_k|d) = \frac{c(t_k, d)}{\sum c(t_i, d)} \tag{2.7}$$

where $c(t_i, d)$ is the frequency of the topic signature t_i in a given document d. The parameters $p(w|t_k)$ can then be estimated by the EM algorithm with the following update formulas:

$$p^n(w) = \frac{(1-\alpha)p^n(w|t_k)}{(1-\alpha)p^n(w|t_k) + \alpha pw|C} \tag{2.8}$$

$$p^{n+1}(w|t_k) = \frac{C(w, D_k)p^n(w)}{\sum_i C(w_i, D_k)p^n(w_i)} \tag{2.9}$$

For each topic signature t_k, we can obtain a set of documents containing the signature this is (D_k), $c(w, D_k)$ is the document frequency of term w in D_k, that is, the cooccurrence count of w and t_k in the whole collection. The final document model for retrieval use is described as follows:

$$p_{bt}(w|d) = (1-\lambda)p_b(w|d) + \lambda p_t(w|d) \tag{2.10}$$

The mapping coefficient λ is to control the influence of two components in the mixture model.

There are two represention of context sensitive topic signatures[4], multiword phrase and ontology concept. Ontology concept needs human knowledge which is not so pratical in use, so we only care about multiword phrase when we use. There are many methods to extract multiword phrase like KPE, LDA-based topic model, Xtract and so on. One can refer to [Zhou Thesis,2008][4] for detail.

References

[1] Jelinek, F. And Mercer, R. *Interpolated estimation of markov sourceparameters from sparse data*, Pattern Recognition in Practice, E. S. Gelsema and L. N. Kanal, Eds., 1980, pp. 381402.

[2] Zhai, C. and Lafferty, J., *A Study of Smoothing Methods for Language Models Applied to Ad hoc Information Retrieval*, In Proceedings of the 24th ACM SIGIR Conference on Research and Development in IR, 2001, pp.334-342.

[3] Zhou, X., Hu, X., Zhang, X., Lin, X., and Song, I.-Y.,*Context-Sensitive Semantic Smoothing for the Language Modeling Approach to Genomic IR*, In the 29th Annual International ACM SIGIR Conference (ACM SIGIR 2006), Aug 6-11, 2006, Seattle, WA, USA

[4] Zhou, X.,: *Zhou Semantics-based language models for information retrieval and text mining.* The Faculty of Drexel University,2008