Document Analysis Report

on US government data set

Hao Zhang

Appropriate TREC measures

MAP is the most appropriate measure for a search system for government web sites.

The reason for this is:

- MAP summarizes the rich information in entire precision-recall curve into a single figure, very efficient regarding evaluation especially in the case we need to measure and compare performance of each topic. Looking at precision-recall graph for every single topic can be time-consuming even impossible.
- For retrieving government documents, its better safe than sorry, factor at all recall levels are more preferable than top results only.
- No fixed recall level or interpolation, easy to adjust configurations to fit into different needs
- Among evaluation measures, MAP has been shown to have especially good discrimination and stability.
- Common and the most standard among the TREC community

Indexing and querying

Lucene did very poorly on MAP on this government documents test collection with the score of only 0.1680, way lower than 0.8~ for the small lab test collection. The average of precision value obtained for recall levels of around 16.8% is not quite optimal.

Summary of performance on MAP over each topic

Good	**Bad**
4. wireless communications	1. mining gold silver coal
14. Agricultural biotechnology	6. physical therapists
18. Shipwrecks	7. cotton industry
19. Cybercrime, internet fraud, and cyber fraud	9. genealogy searches
24. Air Bag Safety	22. Veteran's Benefits
34. Literacy	28. Early Childhood Education
42. homelessness	31. deafness in children
	36. global warming
	37. coin collecting
	39. National Public Radio\/TV
	41. Electric Automobiles
	48. robots

* Good in terms of achieving a MAP ≥ 0.5, bad of MAP = 0

Improving performance

A close look at the summary table above provides some evidences for performance:

- Topics lucene did well are either a single word or very specific and professional terminologies because they are less likely to appear in irrelevant documents
- Topics lucene did poorly are with common words and un-stemmed terms.

Therefore, we can take corresponding techniques to make potential improvement,

- Stemming the term
  This can potentially improve topic 6, 9, 22, 37, 41, 48
- Lemmalization
  This affects poorly performed topic 31, 37, 39
- Search reserved phrase as a whole *
  i.e. implicit AND instead of OR within reserved phrases
  This is the method which I think will lead to the most improvement.
  The idea is phrases like "*coin collecting*" consists of words that is very common when looking at them separately, however their semantic meaning only make sense when grouped as a whole. Lucene currently search them as two separate common terms that is why lots of irrelevant documents are retrieved, if they can be reserved and treated as a single terminology then they will instead fall into the "Good" category – short topics with rare terminology. In this way a poorly performed topic is transformed into a highly likely well performed topic which is a great jump.
  Stakeholders: 6, 7, 9, 36, 39

- Stop word does not seem significant in this case, only topic 31 is potentially affected.

By looking into the content of some of the documents, it seems that these docs were retrieved from websites with URL on top and button/image text here and there. As a result, for the “FIRST_LINE” extracted in DocAdder it will always be an URL which is not what is intended and is meaningless.

Removing the URL and random webpage texts and extract the correct title should help a lot in retrieval accuracy.

By default, the conjunction between terms in topic is OR, it worth a try to make it AND. I suppose each of them work better on specific topics.

Modifications to Lucene

There is a flaw in FileFinder's file search algorithm, it has to get subdirectory recursion enabled to be able to index the gov documents, however in that way it includes index files it generated as well because they lies in a lucene.index subdirectory. Firstly, I fixed this by excluding the index subdirectory.

```
if (recurse && !f.getName().equals("lucene.index"))
```

For term optimization I created a class "ZenOptimizer" which normalizes terms including lower-case, stemming, stop-word removing etc. English Analysis from snowball is used which is the enhancement of standard analyzer with additional EnglishPossesiveFilter, KeywordMarkerFilter and PorterStemFilter.

Core section is cut below,

```
public static String Optimize(String content) throws
IOException {
    StringBuffer result = new StringBuffer();

    if (content != null && content.trim().length() > 0){
        StringReader tReader = new StringReader(content);
        EnglishAnalyzer analyzer = new EnglishAnalyzer();
        TokenStream tStream = null;
        tStream = analyzer.tokenStream("contents", tReader);
        CharTermAttribute term =
tStream.addAttribute(CharTermAttribute.class);

        tStream.reset();
        while (tStream.incrementToken()){
            result.append(term.toString());
            result.append(" ");
        }
        tStream.close();
    }
    System.out.println(result.toString());
    return result.toString();
}
```

In order to get rid of the URL and extract the correct title, below condition is added to DocAdder

```
int keyLinesCounter = 3;
if (keyLinesCounter>0 && !(line.startsWith("http") ||
line.trim().isEmpty())) {
```

to get the first three non-empty and non-URL for weighted later (There are usually back/forth button text in the first two lines).

The ‘title’ field is boosted in the search ranker by altering the query,

```
r.doSearch("(FIRST_LINE:"+optimizedTopic+")^2 OR (CONTENT:"
+ optimizedTopic+")", 5, outputStream);
```

A session of code for enforcing all topic terms appear instead of implicit OR is also added in search ranker for later use to compare performance

```
topicSplited = optimizedTopic.split(" ");
optimizedTopic = topicSplited[0];
for (int i=1; i<topicSplited.length; i++) {
  optimizedTopic += " AND " + topicSplited[i];
}
```

Running the modified Lucene

In order to get a better sense of improvement, I observed the change of performance after each step.

The value of MAP after applying EnglishAnalyzer with stemming, marker/possessive/stop-word filter becomes 0.2079. Slightly better than before. However, most topics performance of which is predicted to be improved after stemming in section 4 remains except for topic 22.

Topic 22. *Veteran's Benefits* --- (refactored-into) ---> *veteran benefits*

It shows that the effect of stemming the possessive is more notable than that of plural.

This aligns with common sense because a possessive twists the word more than plural form and is less likely to appear in contents. The reason filtering plurals does not affect the result is probably if a term appears frequently in a document, its plural will appear frequently as well. Therefore, if the engine was not able to retrieve relevant docs by its plural, it will not perform well on its original form either.

Changing the implicit OR to AND to enforce every term in topic exists remains the same performance. Again I think either way works better for some of the documents, e.g. OR works better for long and verbose queries while AND works better on short precise ones.

Surprisingly, after giving a higher weight for ‘title’ field, the MAP dropped. After consideration I think it is because for this specific data set, documents are retrieved plain text of webpage with navigation button etc. on top so that the top lines do not summarize the paragraph well, even worse than I thought in section 5.

The part I didn't implement is the 'reverse terminology' as I did not find a library for set of terminologies usable in this case. The idea is topics like *Topic 36: global warming* should be searched as a single reserved terminology because otherwise it may look for some documents that talks about e.g. 'Athletes from global is warming up' which is not quite what we want. Terms of these terminologies usually appears together otherwise it means something different.

So that it would be better that there is a terminology-reserver class that detects terminologies in a topic according to pre-defined dictionary, the reserver wraps it inside quotation mark ("global warming" than global warming) to make lucene search it as a whole.