Relevancy Assertion Testing

Two Types of Testing

We can generally talk about two types of relevancy testing:

1. Absolute Truth / Matrix / Grid / TREC / Relevancy Assertions
2. AB Testing / User Preference

Of course these can be further subdivided. Or it could be argued that some tests might have characteristics of both, or tests could be categorized in another way. Most tests seem to fit, at least loosely, into one of these two categories. *Happy to hear your thoughts!*

Relevancy Assertion Testing

The central idea here is that you know what the answers are supposed to be ahead of time. Humans (or some other judge) has compared every test search against every document and answered the question "is this document relevant to this search?", or "how relevant is this document to the search". This set of Relevancy Assurtions can be throught of as an "Absolute Truth" grid or matrix. This is the type of testing TREC focused on.

The main characterics of Relevancy Assertion Testing are:

- A fixed set of test documents, AKA "the Ccorpus"
- A fixed set of test searches that will be run against the test documents, AKA "the Topics"
- A predefined set of judgements made about how well each search matches each document, AKA "the Relevancy Assertions" or "absolute truth"
- The searches will be run by various search enginnes or algorithms over index of documents and the results tabulated.
- Since the correct answers for each search is known ahead of time, the search results from each search engine can be compared against that ideal set of answers.
- Various mathematical formulas are used to compare the actual results to the idea results, and to calculate overall scores.

Contrasting this with "AB Testing"

AB Testing displays the results from two or more search engines and records which search results users prefer. The "A/B" refers to search engine A and search engine B.

There are many variations on this. Modern web sites may show some users results from search engine A while other users see results from engine B, and the site tracks which search engine on average generates more clicks or purchases. More formal testing may show users results from both A and B and ask them to judge which results are more relevant.

The main characteristics of AB Testing are:

- Tracks explicit or implicit preferences between engines A/B
- Statistics heavily relied on to evaluate results, and often to decide which results to present
- A wide variety of implemenations and assesment criteria.
- Users may or may not be aware that they are participating in a test
- Often dispenses with the notion of the "correct" answer; it may be that neither search finds the "best" match.
- May be less labor intensive to setup, since it doesn't require a large predefined corpus of test docs, searches and Relevancy Assertions
- Can be applied to multiple languages or types of content with less additional effort since there is no fixed set of items to translate

This alternative type of testing is discussed here (TODO: link to page once created)

Now... back to Relevancy Assertions!

Types of Relevancy Assertions

Most people think of TREC when they think of this type of testing. Certainly TREC-style assertion testing is important, but it's only one subtype of assertion testing.

Full-Grid Assertions (TREC-Style!)

TREC-style testing is well known and represents one end of a spectrum of Relevancy Assertion Testing, with rigorous data curation and a complete assertion grid. The creation of the relevancy assertion grid itself is also carefully controlled, and the entire grid is considered to have been populated.

But there's an anti-economy of scale with producing a complete grid. As the number of source documents and searches/topics grow, the number of assertions grows geometrically. For M documents and N searches, there are M x N slots in the assertion grid. This adds up very quickly! A very small corpus of 100 documents, evaluated for 30 searches, means 3,000 boxes to fill in. This doesn't sound like much, unless you're the person filling it in! A more reasonable corpus of 2,000 documents and 250 searches would mean a half million potential assertions. This is an absurd amount of "boxes" for a small team to fill in.

- With a giant virtual grid to fill in, techniques can be employed to break up the work into patches; a small subset of 100 documents and 10 searches would give a patch of just 1,000 assertions that one person could potentially fill in. And various techniques could be employed to skip some combination of searches and documents, presuming there are none that are related.

- Another optimization is to cluster a slightly larger set of documents, and then from each cluster delegate a few documents as test searches. Those documents are removed from the corpus, and retasked as searches. However, since they came from a particular cluster, we could assume that those searches are at least moderately relevant to the cluster of docments they were extracted from.
- Or a team could use existing search engines to run the test searches against the corpus and evaluate matching documents deeply down into the result list. So a grader runs the search and notices 4 relevant documents in the first 25 matches. As a check, they could continue scanning the results 10 times further, past document result 250. And then, if no other matches are found, they decide that the original 4 documents in the first 25 results really were the only relevant documents. For thoroughness, perhaps 3 different search engines are used, and the tests are repeated by 2 other graders as well. This is still labor intensive, but not geometrically so.

Using multiple engines and judging the top matches is called "**Pooling**" and is disussed here:

- http://www.ir.uwaterloo.ca/slides/buettcher_reliable_evaluation.pdf

All of these optimization techniques have flaws:

"Flaws" in the sense that it still requires too much, or doesn't insure the absolute best answers have been found.

- Skipping certain patches of documents that are very unlikely to have any relation to a set of searches makes assumptions that may not always be true. Suppose for example I assume that searches related to healthy eating have nothing to do with skiiing. But perhaps there was an article about maintaining good health that estoled the virtues of nutritious food and outdoor winter sports - this document is relevant to both domains and therefore shouldn't be automatically skipped. Perhaps this one examle is considered a fluke. But then another document is found that discusses all of the decadent and healthy food served at various ski lodges. And then a third that talks about winterized insulated canteens, which talks about clean water and specific outdoor activities; perhaps this document is deemed only marginally related eating because water is just "drinking" and very common, and the article mentions many outdoor winter activities and skiiing is only mentioned once in passing. But there's still some relevancy. In a Yes/No grading system perhaps it's decided that's is a "no", but on a percentage basis, the article is still marginally related to healthy eating and should really get at least a 25% relevance. A statistician might argue that these instances are outliers that don't sigficantly affect the overall score, which might be true, but it still diverges from "perfection" of a completely filled in grid.
- Clustering documents, and then repurposing a few documents from each cluster as test searches, just gives a starting point for grading. So searches born from a particular cluster of docs are given a default grade of "B" for all of their siblings left in the cluster, and a default grade of "F" for documents in all other clusters. In this method it's obvious that there will be numerous rading errors. Documents within a cluster likely more relevant to some of their siblings than others. And at least a few of those documents are likely related to documents in other clusters. This becomes even clearer if you recluster the documents, or use a different clustering algorithm. One fix is that humans might still double check the work, perhaps scanning the default grades in some patches, and corrected where needed. But this goes back to the M x N mathematics, although perhaps double checking grades is much faster than creating them from scratch. And there could even be a means of tracking and then predicting which patches need the most attention. But again all these deviate from "perfection".
- The third optimization example mentioned above, using mutliple search engines, and checking down the results well past where the last relevant document was found, also has flaws. For example, since many search engines share well known algorithms, they may tend to make the same mistakes. Perhaps a highly relevant document uses an important term many times, spelled out as 2 words, but the test search combined the term into a single word. So you might have "metadata" vs "meta data" or "doorbell" vs. "door bell". Some modern engines can catch this, espcially if the test search uses the hyphenated form such as "meta-data". This problem is compounded in other languages such as German, where this practice is very common. Or perhaps the test searches were written using American English spellings, whereas a critical document was written by somebody in British English, so many words are a mistmatch, such as color vs. colour, check vs. cheque, etc. Here again modern search engines often support using a thesaurus, but that brings with it other problems. The thesaurus might not be turned on, or if it is turned on, it might actually harm the "precision" measurement (a common problem), or perhaps the open source engine doesn't have a a licensed up-to-date thesaurus for the subject matter being tested. Or perhaps one engine makes one mistake, and the other 2 engines make a different mistake, but they all still miss some critical documents. The point is that this optimization is very likely to miss some matches.

Other forms of Relevancy Assertions

...

Order vs. grade

Detla grading

Web index

domain disambiguation