Full-text search relevance is measured in?

Asked 16 years, 2 months ago Modified 12 years, 2 months ago

Viewed 4k times

16

I am making a quiz system, and when quizmakers insert questions into the Question Bank, I am to check the DB for duplicate / very highly similar questions.

Testing MySQL's MATCH() ... AGAINST(), the highest relevance I get is 30+, when I test against a 100% similar string.

So what exactly is the relevance? To quote the manual:

> Relevance values are non-negative floating-point numbers. Zero relevance means no similarity. Relevance is computed based on the number of words in the row, the number of unique words in that row, the total number of words in the collection, and the number of documents (rows) that contain a particular word.

My problem is how to test the relevance value if a string is a duplicate. If it's 100% duplicate, prevent it from being inserter into Question Bank. But if it is only so similar, prompt the quizmaker to verify, insert or not. So how do I

do that? 30+ for 100% identical string is not percentage, so I'm stump.

Thanks in advance.

mysql performance relevance full-text-search

Share

Improve this question

Follow

edited Oct 20, 2012 at 18:32

hippietrail

16.9k ● 21 ● 109 ● 173

asked Oct 26, 2008 at 12:40

syaz

2,679 ● 6 ● 38 ● 44

3 Answers

Sorted by: Highest score (default)

8

The basic data structure for a text retrieval system is an Inverted Index. This is essentially a list of words found in the document collection with a list of the documents they occur in. It can also have metadata about the occurrence for each document, such as the number of times the word appears.

Documents containing the words can be queried by matching on the search terms. To determine relevance, a heuristic known as a Cosine Ranking is calculated on the hits. This works by constructing n-dimensional vector with one component for each of the n search terms. You can

also weight the search terms if desired. This vector gives a point in n-dimensional space that corresponds to your search terms.

A similar vector based on the weighted occurrences in each document can be constructed from the inverted index with each axis in the vector corresponding with the axis for each search term. If you calculate a dot product of these vectors you get the cosine of the angle between them. 1.0 is equivalent to cos (0), which would assume the vectors occupy a common line from the origin. The closer the vectors together, the smaller the angle and the closer the cosine is to 1.0.

If you sort the search results by the cosine (or bung them into a priority queue as mg does) you get the most relevant. Cleverer relevance algorithms tend to fiddle with the weights of the search terms, skewing the dot product in favour of terms with high relevance.

If you want to dig a little, Managing Gigabytes by Bell and Moffet discusses the internal architecture of text retrieval systems.

Share Improve this answer

Follow

answered Oct 26, 2008 at 14:31

ConcernedOfTunbridge Wells

66.5k ● 15 ● 148 ● 198

andygeers is on the right track: Those numbers have no empirical meaning other than their relations to each other

6

and cannot be used on their own to determine what is or is not an "exact match". You need to determine that yourself. Even aside from the limitations of fulltext search ranking, there's also the open question of just what you consider to consitiute an "exact match". (Actual text only or do soundex matches count? Do synonyms (e.g., "couch" vs. "sofa") count as matching or as distinct? Should an attempt be made to compensate for misspellings? Etc.)

If I had the need to perform such a check, I would grab only the highest-ranked entry returned by the fulltext search, remove any designated stopwords, normalize whitespace, convert to lowercase, do the comparison, and leave it at that until I encountered a case that called for it to be refined further. It's not really all *that* much extra work - if you specify the language you're using for your application, you could probably find someone around here who could write the normalization function within a dozen or so lines of code.

Share Improve this answer

Follow

answered Oct 26, 2008 at 15:47

Dave Sherohman

46.1k ● 14 ● 66 ● 103

2

I don't know the specifics of the MySQL function you're using, but I imagine it could be that there is no absolute meaning for those numbers - they're just designed to be compared with other values produced by the same

function. To check for an absolute match you could select out the text itself and compare manually.

Share Improve this answer

Follow

answered Oct 26, 2008 at 13:11

andygeers

6,966 ● 9 ● 52 ● 63

I prefer to use MySQL search engine whenever possible. If I were to compare my own, I need to do a lot of preparing and checkings e.g. remove all whitespace and special characters, convert all to uppercase, and whatnot. That is my last resort. – syaz Oct 26, 2008 at 13:23