Duplicate Article Matching

Brian Lynnerup Pedersen

Kongens Lyngby 2014

Technical University of Denmark
Department of Applied Mathematics and Computer Science
Matematiktorvet, building 303B,
DK-2800 Kgs. Lyngby, Denmark
Phone +45 4525 3351
compute@compute.dtu.dk
www.compute.dtu.dk

Summary (English)

The goal of the thesis is to - snaps

Summary (Danish)

Målet for denne afhandling er at lave noget forskelligt

Preface

This thesis was prepared at the department of Informatics and Mathematical Modelling at the Technical University of Denmark in fulfilment of the requirements for acquiring an B.Eng. in Informatics.
The project is equal to 20 ECTS points.

The thesis deals with ...

The thesis consists of ...

Lyngby, 09-June-2014

Brian Lynnerup Pedersen

Acknowledgements

I would like to thank my supervisors from DTU, Inge Li Gørtz and Philip Bille, for the help they provided to my project. Also I would like to thank my company Infomedia, for letting me do my project with them, my project leader Klaus Wenzel Jørgensen and Rene Madsen.

Contents

Chapter 1

Introduction

blah blah blah stuff about Infomedia...

blah blah blah general stuff (what is a "term" "article" "vector" and so on...)

Chapter 2

Algorithms in General

Considerations regarding pick of algorithms.

Bag of Words (Cosine), generates a vector from each document. Another document can then be matched with the list of documents, and their vectors are compared based on the dot product. The closer the dot product are to the value 1.0 the more similar are two documents. A score of 0.0 indicates that two words has nothing in common.

Problem: This string comparison is very sensitive to short articles, for instance "A man walks his dog in the park" and "A dog walks his man in the park" would result in returning a cosine value of 1.0, due to the term vector. This problem is very unlikely to yield false positives.

Longest Common Substring (LCS), compares documents in pairs. A general implementation would be to have list of documents and then compare a document to each of the documents in the list. The algorithm will then return the length of the longest common substring. This would probably needed to be modified to return a percentage match in stead (total string length / longest common substring found).

Problem: This algorithm is very prone to fail in cases where there have been made alterations to the article in question. A word change in the middle of

one of two otherwise identical articles will result in a 50 percent match. If the article in question have been obfuscated with many changed words, the LCS will be extremely short. This is a high risk problem, as article duplication will often involve changing words.

2.1 Optimizing Performance

2.1.1 Stop Words

Stop word removal would improve running time (performance) of both algorithms, and would pose little threat of causing either algorithm to fail. The exception to this could be very short articles (like breaking news articles), that only contains common words, like "Man walks away". Depending on the stop word list, this article could end up being *null*. We can safely (with respects to the previously addressed problem) remove stop words, as they don't provide any *semantic* value to the text.

2.1.1.1 Cosine

For the cosine algorithm removal of stop words would improve performance, by reducing the size of the *Magnitude Vector*.

2.1.1.2 LCS

In regards to the LCS algorithm, the performance would also be improved, as the algorithm wouldn't need to traverse as many *terms*. This would reduce the length of the longest common substring, but it would be unlikely that it would affect the outcome of the algorithm.

2.1.2 Stemming

Stemming would can improve performance of the cosine algorithm. This would not affect accuracy as stemming does not ruin the semantic content.

2.1.2.1 Cosine

Stemming improves the performance of the cosine algorithm as this will reduce the number of words in the *vector space*. As words would be reduced to their base form (Danish: *Grundform*), words used several times, but with different endings would be counted as the same word. When using weighed evaluation this would actually improve performance.

2.1.2.2 LCS

Stemmning would not improve the performance of the LCS algorithm. As this algorithm matches terms one by one, it would make no difference if the words are stemmed or not. For LCS it would actually decrease performance to apply stemming, as this operation would consume time while preparing the files, with no impact of the algorithms actual performance.

2.2 Implementation of Stop Words and Stemming

For the algorithms tested removing stop words would be an improvement (**TEST!!!**), and would be valuable to do before using either algorithm.

Stemming would should not be done when using the LCS algorithm as that has no performance enhancing impact on the LCS.

Appendix A

Appendix

This appendix is full of stuff ... add more

Document from Infomedia about the general Vector Spaced Search.
Thesis document from Inge
Links to Wikipedia
The algorithm course book..