University of Sheffield

Document Retrieva

Guoyu Liu

Supervisor: Professor Robert Gaizauskas and Temitope Adeosun

November 10, 2022

1 Description

1.1 Introduction

The three weights are calculated by returning 10 relevant documents based on the weight score from highest to lowest given query.Time() cannot effect the result.

1.2 three model

This core is computed by this function $\frac{\sum_{i=1}^{n} q_i d_i}{\sqrt{\sum_{i=1}^{n} d_i^2}}$,In the Binary mode, q_i =0 if the term is not in the given query, and $q_i = 1$ if the term in the query. $q_i d_i$ is represents the number of term in the query and in the document. The length of the documents is represented by $\sum_{i=1}^{n} d_1^2$

Tf model is computed by this function $\frac{\sum_{i=1}^{n} q_i d_i}{\sqrt{\sum_{i=1}^{n} d_i^2}}$,In the TF mode,q_i is the frequency of the term in the given query and d_i is the frequency of the same term in the document. The score is higher if the term appear in given query and document.($\sqrt{\sum_{i=1}^{n} d_i^2}$) is the weighted sum of all the words in that document.

TFIDF mode is computed by this function $\frac{\sum_{i=1}^{n} q_i d_i}{\sqrt{\sum_{i=1}^{n} d_i^2}}$,In the TFIDF mode, idf$_i$ =$\log_{10} \frac{|D|}{d_i}$,In the code the query's and every document's tf ,idf and tfidf are calculated before the final function run . In order to reduce the time ,for the calculated the query part will use the relevant document to do it. we care about the top 10 highest score documents.

2 Results

No stoplist, **No** stemming			
	Binary	TF	TFIDF
Rel_Retr	19	49	110
Precision	0.03	0.08	0.17
Recall	0.02	0.06	0.14
F-measure	0.03	0.07	0.15
Time (sec.)	2.73	16.83	20.90

No stoplist, **With** stemming			
	Binary	TF	TFIDF
Rel_Retr	22	73	154
Precision	0.03	0.11	0.24
Recall	0.03	0.09	0.19
F-measure	0.03	0.10	0.21
Time (sec.)	2.60	18.02	20.60

With stoplist, **No** stemming			
	Binary	TF	TFIDF
Rel_Retr	47	107	119
Precision	0.07	0.17	0.19
Recall	0.06	0.13	0.15
F-measure	0.07	0.15	0.17
Time (sec.)	1.81	6.04	7.80

With stoplist, **With** stemming			
	Binary	TF	TFIDF
Rel_Retr	34	122	172
Precision	0.05	0.19	0.25
Recall	0.04	0.15	0.20
F-measure	0.05	0.17	0.23
Time (sec.)	0.96	9.65	10.01

Table 1: Results of three weighting schemes

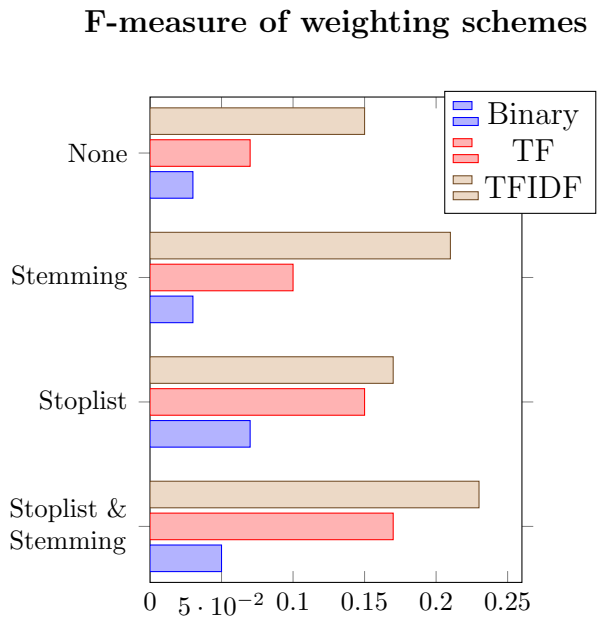

Figure 1: F-measures

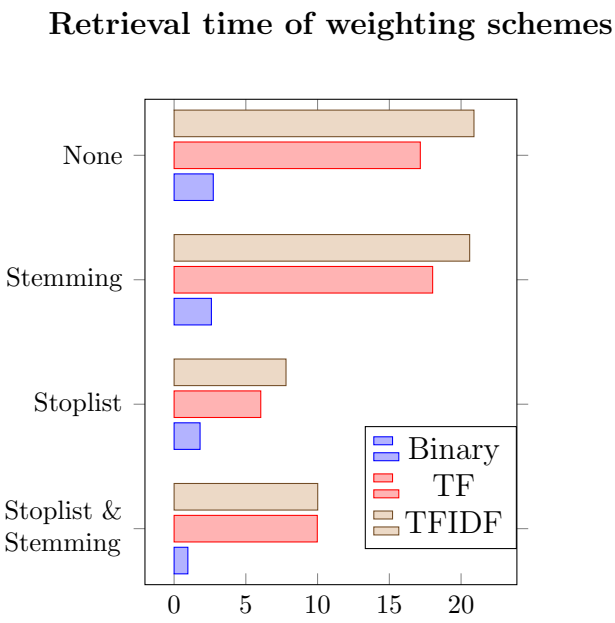

Figure 2: Retrieval time

3 Conclusion

From table 1, it can be seen that the overall performance of the TFIDF weighting scheme is the best, and the performance of the Binary weighting scheme is the worst, but in the second table, it can be seen that the Binary weighting scheme has the shortest time, but the longest time is TFIDF weighting scheme. According to the table1 data, the weighting scheme of TFIDF has small changes in data after using stemming, but has been greatly improved after using stoplist. As can be seen from the colored chart below, the time will be longer after using stoplist and stemming, because the preprocessing takes more time.

It is clear that the TFIDFmodel data did not change much after using stoplist, but improved dramatically after using stemming. The other two models did the opposite. In my opinion, it is possible that the TFIDF scheme may have reduced the total number of words by lyricizing the words in the file and query.

From the color icon, it can be seen that the time of -s detection will be greatly reduced compared with that of -p detection. This is because more cycles and calculations will be added or reduced during retrieval due to the use of stem, which leads to more time after stoplist is used. In addition, I tried to initialize some calculations as much as possible when experimenting with the method, so as to reduce the repetition of the calculation cycle (although I tried to do this, the final time was not optimal).

overall, the IR system can be improved by using stoplist and stemming, and the efficiency of this system is mainly affected by experimental methods, while the methods are affected by different algorithms of each person. Initializing a good portion of the code prevents code refactoring and reduces the extra time consumed by unnecessary loops. In the end, I think the TFIDF model makes the most sense when it comes to both time and efficiency