Data Mining - HW 1

Evaluation of Search Engines and Near Duplicates Detection

Simone Marretta 1911358 Luca Scofano 1762509

1 Part 1 - Search Engines evaluation

1.1 Description of all Search Engines

Remark: We used **Field booster** and **Multifield Parsers** only for The ***Cranfield Dataset*** since 'Title' field will have an impact on the scoring as well

Table 1: Description of the Analyzers

Type	**Description**
Field Booster	It gives a higher (x1.5) score if the query is present in a certain field (Title in our case).
Stemming	Composes a RegexTokenizer with a lower case filter, an optional stop filter, and a stemming filter.
Simple Analyzer	Composes a RegexTokenizer with a LowercaseFilter.
Standard	Composes a RegexTokenizer with a LowercaseFilter and optional StopFilter.
Ngram	Composes an NgramTokenizer and a LowercaseFilter.

Table 2: Description of the Scoring functions

Number	**Type**	**Description**
1	BM25F	It ranks a set of documents based on the query terms appearing in each document, regardless of their proximity within the document.
2	Frequency	It computes the frequency of a term in the document.
3	PL2	Scoring function created by Terrier, an open source search engine.
4	TFIDF	Based on the TFIDF scoring method, where Term Frequency and Inverse Document Frequency are computed.
5	posScore	It score documents based on the earliest position of the query term in the document. (Whoosh's documentation)

1.2 Number of indexed documents and number of queries

The assignment has to be carried out on two different data-sets, one is called *Cranfield* and the other one *Time*. *Doc* = Number of documents, *Queries* = Number of queries, *GT* = Number of queries in Ground Truth.

$$\textbf{\textit{Cranfield}}: \begin{cases} Doc = 423 \\ Queries = 225 \\ GT = 110 \end{cases} \qquad \textbf{\textit{Time}}: \begin{cases} Doc = 1.400 \\ Queries = \quad 83 \\ GT = \quad 80 \end{cases}$$

1.3 MRR Table for all Search Engines

The Mean Reciprocal Rank or ***MRR*** is not defined on a single query but it's defined on a group of queries.

$$MRR(Q) = \frac{1}{|Q|} \sum_{\forall q \epsilon Q}^{Q} \frac{1}{index(FirstRelevantResult)}$$

Weak spot: We are loosing information on how many relevant results are shown in top k positions, it only shows the First relevant Result.

Figure 1: Cranfield dataset

Search Engine	Score
se_1_Standard.csv	0.517755
se_3_Stemming.csv	0.508782
se_1_Stemming.csv	0.507855
se_3_Standard.csv	0.482964
se_1_Field Booster.csv	0.476545
se_1_Simple Analyzer.csv	0.474518
se_3_Ngram.csv	0.439600
se_1_Ngram.csv	0.429418
se_3_Field Booster.csv	0.422591
se_4_Stemming.csv	0.404618
se_3_Simple Analyzer.csv	0.404400
se_4_Standard.csv	0.387827

(a) Top 12

Search Engine	Score
se_2_Stemming.csv	0.320245
se_2_Standard.csv	0.313791
se_5_Stemming.csv	0.310600
se_5_Standard.csv	0.288064
se_4_Ngram.csv	0.230400
se_5_Simple Analyzer.csv	0.216764
se_5_Field Booster.csv	0.216764
se_5_Ngram.csv	0.196636
se_4_Field Booster.csv	0.194545
se_2_Ngram.csv	0.169164
se_4_Simple Analyzer.csv	0.165527
se_2_Field Booster.csv	0.065264
se_2_Simple Analyzer.csv	0.058727

(b) Last 13

Figure 2: Search engines based on MRR score

Figure 3: Time Dataset

Search Engine	Score
se_1_Stemming.csv	0.696300
se_1_Standard.csv	0.679250
se_1_Simple Analyzer.csv	0.668588
se_3_Stemming.csv	0.635225
se_3_Standard.csv	0.613538
se_1_Ngram.csv	0.558588
se_4_Standard.csv	0.536425
se_4_Stemming.csv	0.502863
se_2_Standard.csv	0.460538
se_5_Standard.csv	0.458325

(a) Top 10

Search Engine	Score
se_5_Stemming.csv	0.455750
se_2_Stemming.csv	0.425400
se_3_Ngram.csv	0.419625
se_5_Ngram.csv	0.387600
se_3_Simple Analyzer.csv	0.309387
se_5_Simple Analyzer.csv	0.265912
se_4_Ngram.csv	0.257400
se_4_Simple Analyzer.csv	0.240263
se_2_Ngram.csv	0.214600
se_2_Simple Analyzer.csv	0.154362

(b) Last 10

Figure 4: Search engines based on MRR score

1.3.1 MRR Table for the top 5 Search Engines

Figure 5: Top 5 Search Engines based on MRR

Search Engine	Score
se_1_Standard.csv	0.517755
se_3_Stemming.csv	0.508782
se_1_Stemming.csv	0.507855
se_3_Standard.csv	0.482964
se_1_Field Booster.csv	0.476545

(a) Cranfield

Search Engine	Score
se_1_Stemming.csv	0.696300
se_1_Standard.csv	0.679250
se_1_Simple Analyzer.csv	0.668588
se_3_Stemming.csv	0.635225
se_3_Standard.csv	0.613538

(b) Time

1.4 R Precision Table

When: ***k = GT(q)*** then ***P@k = R precision***

$$R - precision = \frac{NumberOfRelevantDocumentsInFirst|GT(q)|positions}{GT(q)}$$

SE Cofiguration	**Mean**	**Min**	**1st quartile**	**Median**	**3rd quartile**	**Max**
se_1_Standard.csv	0.257713	0.0	0.0	0.25	0.428571	1.0
se_3_Stemming.csv	0.264101	0.0	0.0	0.25	0.428571	1.0
se_1_Stemming.csv	0.265212	0.0	0.0	0.25	0.428571	1.0
se_3_Standard.csv	0.257402	0.0	0.0	0.25	0.440476	1.0
se_1_Field Booster.csv	0.244790	0.0	0.0	0.25	0.421429	1.0

Figure 6: Cranfield

SE Cofiguration	**Mean**	**Min**	**1st quartile**	**Median**	**3rd quartile**	**Max**
se_1_Stemming.csv	0.527260	0.0	0.191667	0.5	0.888889	1.0
se_1_Standard.csv	0.547587	0.0	0.321429	0.5	0.888889	1.0
se_1_Simple Analyzer.csv	0.542006	0.0	0.321429	0.5	0.888889	1.0
se_3_Stemming.csv	0.482691	0.0	0.000000	0.5	0.808333	1.0
se_3_Standard.csv	0.475147	0.0	0.000000	0.5	0.808333	1.0

Figure 7: Time

Figure 8: R-precision table of Top 5 Search engines based on MRR score

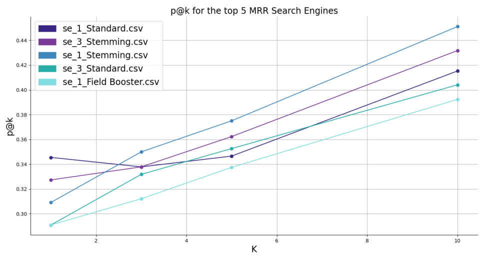
1.5 P@k Plot

$$P@k = \frac{NumberOfRelevantDocumentsInFirstKpositions}{k}$$

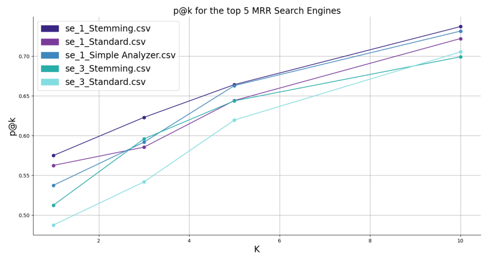
numerator = is the number of relevant documents in the Ground truth that are in the first k positions of the result based on the score. *denominator* = number of k top documents we picked.

Can we get a better denominator, in this case we can't get 1 as a perfect score? ***YES***, the ***normalized*** score, and this is what we've used in the homework. New formula:

$$P@k = \frac{NumberOfRelevantDocumentsInFirstKpositions}{min(k, GroundTruth(q))}$$



(a) Cranfield



(b) Time

Figure 9: p@k plot of Top 5 Search engines based on MRR score

1.6 nDCG@k plot

This evaluation metrics tries to account for all the weaknesses of the previous evalution systems.

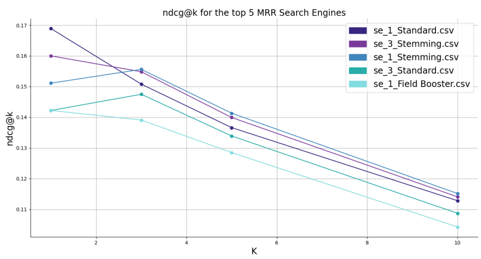
$$nDCG(query, k) = \frac{DCG(query, k)}{IDCG(query, k)}$$

IDCG is the ideal discounted cumulative gain and it's a DCG of a perfect ranking algorithm.

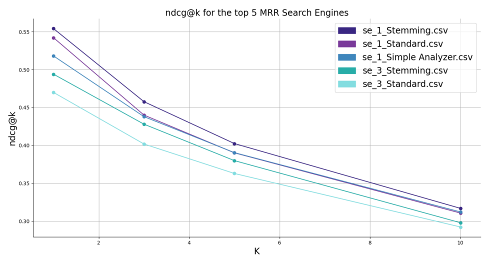
Components of the formula:

$$DCG(query, k) = \sum_{p=1}^{k} \frac{relevance(docID, query)}{log_2(position + 1)}$$

What is the **relevance**? *relevance(docID, query) = 1* if docID belongs to $GT(q)$ and 0 otherwise



(a) Cranfield



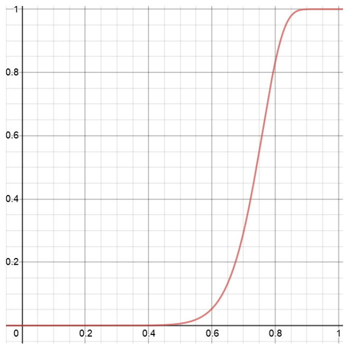
(b) Time

Figure 10: nDCG@k plot of Top 5 Search engines based on MRR score

2 Part 2 - Near Duplicate detection

2.1 Number of row and number of bands

The *Number of rows are* ***12*** and the *Number of bands are* ***25***



2.2 The probability to have False-Negatives, in the set of candidate pairs, for the following Jaccard values: 0.89, 0.9, 0.95 and 1

Probability to have false-negatives is :

$$(1 - J(a,b)^r)^b$$

Probability to have false-negatives for ***0.89*** *:*

$$(1 - 0.89^r)^b = 0.00083$$

Probability to have false-negatives for ***0.9****:*

$$(1 - 0.9^r)^b = 0.00025$$

Probability to have false-negatives for ***0.95****:*

$$(1 - 0.95^r)^b = 3.63449e - 09$$

Probability to have false-negatives 1 :

$$(1 - 1^r)^b = 0$$

2.3 The probability to have False-Positives, in the set of candidate pairs,for the following Jaccard values: 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55 and 0.5

Jaccard Similarity	Probability to have False Positives
0.85	0.97842
0.8	0.83134
0.75	0.55279
0.7	0.29422
0.65	0.13290
0.6	0.05302
0.55	0.01898
0.5	0.00608

2.4 How did you reduce the probability to have False-Negatives?

We reduce the probability to have False Negatives choosing r and b so that the threshold was significantly inferior to 0.89. We know from the Mining of Massive Datasets book that the threshold is approximately

$$(1/b)^{(1/r)}$$

. **We chose r and b so that:**

$$(1/b)^{(1/r)} = 0.765$$

2.5 The Execution-Time of the Near-Duplicates-Detection tool

The execution time was **4 minutes and 8 seconds.**

2.6 The number of Near-Duplicates couples you found

The number of Near-Duplicates was **39697**.

2.7 The number of Near-Duplicates couples you found with an approximated Jaccard similarity value of at least 0.89, 0.90, 0.91, 0.92, 0.93, 0.94, 0.95, 0.96, 0.97, 0.98, 0.99, 1

Jaccard Similarity	Number of Near Duplicates couples
0.89	39697
0.90	38773
0.91	37849
0.92	36961
0.93	36150
0.94	35513
0.95	34466
0.96	33786
0.97	33081
0.98	32428
0.99	31920
1	31691