CS 6501: Information Retrieval

MP3 – Retrieval Functions

Yiwei Fang
yf5kq

11/30/2018

1. **Copy and paste your implementation of each ranking algorithm, together with the corresponding final MAP/P@10/MRR/NDCG@10 performance you get from each ranking function. *Use the default parameter settings suggested here* (30pts)**

   a) Boolean

```java
package edu.virginia.cs.index.similarities;

import org.apache.lucene.search.similarities.BasicStats;

public class BooleanDotProduct extends SimilarityBase {
    /**
     * Returns a score for a single term in the document.
     *
     * @param stats
     *            Provides access to corpus-level statistics
     * @param termFreq
     * @param docLength
     */
    @Override
    protected float score(BasicStats stats, float termFreq, float docLength) {
        return termFreq > 0 ? 1 : 0;
    }

    @Override
    public String toString() {
        return "Boolean Dot Product";
    }
}
```

```
Boolean Models

MAP: 0.1765941441732281

P@10: 0.2881720430107527

MRR: 0.5943108713270008

NDCG: 0.34981023431355174
```

   b) tf-idf

```java
package edu.virginia.cs.index.similarities;

import org.apache.lucene.search.similarities.BasicStats;

public class TFIDFDotProduct extends SimilarityBase {
    /**
     * Returns a score for a single term in the document.
     *
     * @param stats
     *            Provides access to corpus-level statistics
     * @param termFreq
     * @param docLength
     */
    @Override
    protected float score(BasicStats stats, float termFreq, float docLength) {

        double logTF = 1 + Math.log(termFreq) / Math.log(2);
        double logDF = Math.log((stats.getNumberOfDocuments()+1)/stats.getDocFreq()) / Math.log(2);
        float score = (float) (logTF*logDF);
        return termFreq > 0 ? score : 0;
    }

    @Override
    public String toString() {
        return "TF-IDF Dot Product";
    }
}
```

TF-IDF Dot Product

MAP: 0.18216935540724435

P@10: 0.2870967741935484

MRR: 0.6403732586725946

NDCG: 0.36184649301986205

c) bm25

```java
     * @param stats
     *            Provides access to corpus-level statistics
     * @param termFreq
     * @param docLength
     */
    @Override
    protected float score(BasicStats stats, float termFreq, float docLength) {

        System.out.println("termFreq " + termFreq + "docLength " + docLength + "getNumberDocs " + stats.getNumberOfDocume
        double k1 = tuneParam.k1;
        double k2 = tuneParam.k2;
        double b = tuneParam.b;

        double termOne = Math.log((stats.getNumberOfDocuments() - stats.getDocFreq() + 0.5) / (stats.getDocFreq() + 0.5))
        double termTwo = ((k1+1)*termFreq) / (k1*(1-b+(b*docLength/stats.getAvgFieldLength())) + termFreq);
        double termThree = ((k2+1)*1) / (k2+1);

        float score = (float) (termOne*termTwo*termThree);
        return termFreq > 0 ? score : 0;
    }

    @Override
    public String toString() {
        return "Okapi BM25";
    }
```

Okapi BM25

MAP: 0.1867936950148781

P@10: 0.30752688172043013

MRR: 0.5933650504215023

NDCG: 0.3681466348357197

d) pivoted length

```java
public class PivotedLength extends SimilarityBase {
    /**
     * Returns a score for a single term in the document.
     *
     * @param stats
     *          Provides access to corpus-level statistics
     * @param termFreq
     * @param docLength
     */
    @Override
    protected float score(BasicStats stats, float termFreq, float docLength) {
        double s = 0.75;

        double termOne = (1 + Math.log(1 + Math.log(termFreq)))/(1 - s + s*docLength/stats.getAvgFieldLength());
        double termTwo = 1;
        double termThree = Math.log((stats.getNumberOfDocuments()+1) / stats.getDocFreq());

        float score = (float) (termOne*termTwo*termThree);
        return termFreq > 0 ? score : 0;
    }

    @Override
    public String toString() {
        return "Pivoted Length Normalization";
    }

}
```

Pivoted Length Normalization

MAP: 0.12913287570603849

P@10: 0.23978494623655922

MRR: 0.43543198402547073

NDCG: 0.27464230861257105

e) Jelinek-Mercer

```
     * @param stats
     *            Provides access to corpus-level statistics
     * @param termFreq
     * @param docLength
     */
    @Override
    protected float score(BasicStats stats, float termFreq, float docLength) {
        double lamda = 0.1;
        double alpha = lamda;
        double pwc = model.computeProbability(stats);
        double psmooth = (1-lamda)*(termFreq/docLength) + lamda*pwc;

        double termOne = Math.log(psmooth/(alpha*pwc)) / Math.log(2);
      //|q|log(alpha) ignored
        float score = (float)(termOne);

        return termFreq > 0 ? score : 0;
    }

    @Override
    public String toString() {
        return getName();
    }
```

Jelinek-Mercer Smoothing

MAP: 0.2257331643786167

P@10: 0.34193548387096767

MRR: 0.6781211872672974

NDCG: 0.4201580145765151

f) Dirichlet Prior

```
     * @param stats
     *            Provides access to corpus-level statistics
     * @param termFreq
     * @param docLength
     */
    @Override
    protected float score(BasicStats stats, float termFreq, float docLength) {

        double miu = tuneParam.miu;
        double alpha = miu / (miu + docLength);
        double pwc = model.computeProbability(stats);
        double psmooth = (termFreq + miu*pwc)/(docLength+miu);

        double termOne = Math.log(psmooth/(alpha*pwc)) / Math.log(2);
        //|q|log(alpha) ignored
        float score = (float)(termOne);

        return termFreq > 0 ? score : 0;
    }

    @Override
    public String getName() {
        return "Dirichlet Prior";
    }
```

Dirichlet Prior Smoothing

MAP: 0.14505872136683184

P@10: 0.2376344086021506

MRR: 0.5348695059217692

NDCG: 0.2885881563076047

2. **Please carefully tune the parameters in BM25 and Dirichlet prior smoothed Language Model. Report the best MAP you have achieved and corresponding parameter settings. (20pts)**
Designed a class called tuneParam.class which contains the default values of k1, k2, b and miu.

```
package runners;

public class tuneParam {
    public static double k1 = 1.5;
    public static double k2 = 750;
    public static double b = 1.0;
    public static double miu = 2500;
}
```

a) BM25

```
Okapi BM25

max MAP of BM25: 0.23022684043939504

k1 is at: 1.2

b is at: 0.75
```

b) Dirichlet prior

```
Dirichlet Prior Smoothing

max MAP of Dirichlet Prior: 0.14976987988046905

miu is at: 2000.0
```

3. **With the default document analyzer, choose one or two queries, where Pivoted Length Normalization model performed significantly better than BM25 model in average precision, and analyze what is the major reason for such improvement? Perform the same analysis for Pivoted Length Normalization v.s. Dirichlet Prior smoothed Language Model, and BM25 v.s. Dirichlet Prior smoothed Language Model, and report your corresponding analysis (using your best parameters for BM25 and Dirichlet Prior smoothed Language Model). (20pts)**

a) Pivoted Length Normalization model vs. BM25 model and analyze what is the major reason for such improvement.

```
0.003758842783233027
-0.07876899455846822
-0.006780538302277428
-0.0020444435489615914
-0.06906210760166134
-0.0668771990567656
-0.01995824165630835
-0.04099919970822579

pl ap: 0.17023809523809522

bm25 ap: 0.13669467787114847

Max AP difference: 0.03354341736694674
Query: 56
```

Query 56: the synthesis of networks with given sampled data transfer functions

BM25:

```
Query: the synthesis of networks with given sampled data transfer functions
X 0. 7697
  1. 4412
X 2. 4204
  3. 6138
X 4. 9851
X 5. 6448
X 6. 6970
X 7. 9692
X 8. 10336
X 9. 9796
X 10. 4074
X 11. 6782
X 12. 782
X 13. 7695
X 14. 7173
X 15. 2192
  16. 1049
X 17. 5973
```

Pivoted Length:

```
Query: the synthesis of networks with given sampled data transfer functions
X 0. 7697
  1. 4412
  2. 6138
X 3. 6448
X 4. 4204
X 5. 7762
X 6. 8002
X 7. 9851
X 8. 10045
X 9. 9692
X 10. 782
  11. 4614
X 12. 8820
  13. 1049
X 14. 2192
X 15. 7088
X 16. 7341
X 17. 7543
```

4412: the pulse transfer function and its application to sampling servo systems
6138: the pulse transfer function and its application to sampling servo systems discussion on
4204: optimum network functions for the sampling of signals in noise transfer functions are calculated for networks which maximize the ratio between the average amplitude of n successive samples of the output signal and the rms output noise and a continuous sample of the output
7697: synthesis of the transfer function of terminal pair networks

Pivoted Length Normalization and BM25 perform almost similar to each other when the documents length is at the average level, like doc 4412, 6138 and 7697. While for the document 4202, BM25 has the higher rank than the Pivoted Length Normalization. To analyze the reason behind it, I think it maybe the different normalization method for document length Pivoted Length Normalization use. For pivoted length normalization, the normalizer =1 if the docLength = average doc length. So, it performs better than BM25 on long documents, such as 4204.

Pivoted Length normalizer: $1 - s + s\frac{n}{n_{avg}}$ = 1 when n = navg.

b) Pivoted Length Normalization vs. Dirichlet Prior smoothed Language Model

```
0.0464890994908875
-0.07985229179391057
-0.001872648719330643
-0.05013064892097149
-0.017135762958679142
0.08624890651781404
0.1413373860182371
-0.018111286753844904

pl ap: 0.3333333333333333

dp ap: 1.0

Max AP difference: 0.6666666666666667
Query: 7
```

Query 7+1 = 8: measurement of plasma temperatures in arc discharge using shock wave techniques

Pivoted Length:

```
Query: measurement of plasma temperatures in arc discharge using shock wave techniques
X 0. 11350
X 1. 1998
  2. 3774
X 3. 191
X 4. 58
X 5. 9588
X 6. 10788
X 7. 4117
X 8. 987
X 9. 672
X 10. 978
X 11. 694
X 12. 8051
X 13. 7935
X 14. 7937
X 15. 3316
X 16. 10912
```

Dirichlet Prior:

```
Query: measurement of plasma temperatures in arc discharge using shock wave techniques
  0. 3774
X 1. 11350
X 2. 6300
X 3. 8849
X 4. 11130
X 5. 10788
X 6. 2125
X 7. 4117
X 8. 7888
X 9. 4028
X 10. 8617
X 11. 708
X 12. 4810
X 13. 4753
X 14. 7931
X 15. 4702
```

11350: x band measurement of shock tube plasma temperature the radiation temperature of a plasma behind a hypersonic shock wave is measured using a microwave receiver results indicate that the temperature is approximately
1998: oblique shock waves in a plasma with finite conductivity
3774: structure of shock waves in a plasma investigation of a shock wave in a plasma taking account of the difference in electron and ion temperatures three cases are examined nonstationary shock wave stationary shock wave in a strong magnetic field

For the query 8, Dirichlet Prior performs significantly better than the Pivoted length normalization, because according to the formula, for short queries, the Dirichlet smoothing

performs better than the longer document. In the other words, the longer the document, the less smoothing is applied. It makes sense that the smoothing function will make doc 3774 rank higher by re-allocating the extra counts such that unseen words will have a non-zero count. While for vector space model, without smoothing, the unseen words will degrade the cosine similarity values between query vector and doc vector.

c) BM25 model vs. Dirichlet Prior smoothed Language Model

```
-0.008653187021608071
-0.052175092469933085
-0.08619787056034048
0.019371707461048437
0.12137914436192873
-0.05911048646207069

dp ap: 1.0

bm25 ap: 0.5

Max AP difference: 0.5
Query: 7
```

Query 7: measurement of plasma temperatures in arc discharge using shock wave techniques

BM25:

```
Query: measurement of plasma temperatures in arc discharge using shock wave techniques
X 0. 11350
  1. 3774
X 2. 1998
X 3. 10788
X 4. 9588
X 5. 191
X 6. 58
X 7. 10912
X 8. 9209
X 9. 987
X 10. 4117
X 11. 978
X 12. 672
X 13. 8051
X 14. 7935
X 15. 7937
X 16. 5083
```

Dirichlet Prior:

```
Query: measurement of plasma temperatures in arc discharge using shock wave techniques
  0. 3774
X 1. 11350
X 2. 6300
X 3. 8849
X 4. 11130
X 5. 10788
X 6. 2125
X 7. 4117
X 8. 7888
X 9. 4028
X 10. 8617
X 11. 708
X 12. 4810
X 13. 4753
X 14. 7931
X 15. 4702
```

11350: x band measurement of shock tube plasma temperature the radiation temperature of a plasma behind a hypersonic shock wave is measured using a microwave receiver results indicate that the temperature is approximately

As we have discussed in above, the language performs better on this query than the vector space model because of the Dirichlet prior. And for this same query, BM25 and Pivoted ranked top three docs are different because of pivoted length normalization as I discussed in subproblem a.

4. Pick one of the previously implemented scoring functions out of

a) Okapi BM25

b) Pivoted Length Normalization

c) Language Model with Dirichlet Smoothing

to analyze under what circumstance the chosen scoring function will mistakenly favor some less relevant document (*i.e.*, ranks a less relevant document at a higher position than a more relevant one). Please correspond your analysis with what you have found in Problem 3.

After reading the paper An Exploration of Axiomatic Approaches to Information Retrieval, 1) can you briefly summarize the major contribution of this paper? 2) how do you think you can fix the problem you have identified in the ranking result analysis? Please relate your solution and corresponding implementation in the report. Also report the resulting ranking performance of your revised ranking algorithm. (30pts)

Summary:

1) The paper proposes an axiomatic framework with two important components: function space and retrieval constraints by experimenting the optimized scoring functions on three representative retrieval functions: PN, Okapi and DP within axiomatic framework, and achieved great progress. The biggest contribution of this paper is about the introduced framework consisting of an inductive scheme for any scoring function definitions and formalized retrieval constrains. This paper demonstrates a stable way to derive new retrieval functions using three basic constraints: primitive weighting function, query growth function and document growth function. The paper also indicates that with more reasonable axiomatic constraints, the derived functions will be more specialized and performing better.

2) Optimized Pivoted Length Normalization:

```
     * @param termFreq
     * @param docLength
     */
    @Override
    protected float score(BasicStats stats, float termFreq, float docLength) {
        double sp = 0.1;

//      double termOne = (1 + Math.log(1 + Math.log(termFreq)))/(1 - s + s*docLength/stats.getAvgFieldLength());
//      double termTwo = 1;
//      double termThree = Math.log((stats.getNumberOfDocuments()+1) / stats.getDocFreq());
//

        double s = sp / (1-sp);
        double termOne = 1 + Math.log(1 + Math.log(termFreq));
        double termTwo = 1;
        double weight = Math.log((stats.getNumberOfDocuments() + 1) / stats.getDocFreq());
        double termFour = (stats.getAvgFieldLength() + s)/(stats.getAvgFieldLength() + docLength*s);

        float score = (float) (termOne*termTwo*weight*termFour);
        return termFreq > 0 ? score : 0;
    }
```

Modified Pivoted Length Normalization:

```
MAP: 0.23239101279448654

P@10: 0.35591397849462375

MRR: 0.6836004049713729

NDCG: 0.43310780677552896
```

Original Pivoted Length Normalization:

```
Pivoted Length Normalization

MAP: 0.12913287570603849

P@10: 0.23978494623655922

MRR: 0.43543198402547073

NDCG: 0.27464230861257105


0.039161490683229806
-0.036608745100154626
0.047936472338095046
0.12363218012260167
0.0984641556450067
0.1139413347559578

pl ap: 0.5170454545454546

pl_old ap: 0.015625

Max AP difference: 0.5014204545454546
Query: 64
```

Query: the use of complex variables in the theory of communication networks

```
Query: the use of complex variables in the theory of communication networks
  0. 5470
  1. 6783
X 2. 21
X 3. 7562
X 4. 6764
X 5. 4836
X 6. 8118
X 7. 3392
X 8. 4828
X 9. 7280
X 10. 7695
```

```
Query: the use of complex variables in the theory of communication networks
X 0. 10629
X 1. 2268
X 2. 2960
X 3. 3463
X 4. 10702
X 5. 2224
X 6. 4244
X 7. 3501
X 8. 10169
X 9. 8227
```

5470: application of complex symbolism to linear variable networks the theory developed is based on frequency domain analysis equations are derived for networks containing one linear variable element the value of which varies periodically and is capable of being developed in a fourier series the usefulness of these equations for computations of magnetic and dielectric modulators and amplifiers is indicated
10629: an adaptive communications filter a filter is described which is optimum for a completely random communication signal

From the results of valuation methods P@K, MRR, MAP and NDCG between old PN and axiomatic PN, we can notice that there is a big improvement of the scoring function of the axiomatic PN. By understanding the axiomatic framework, we can better apply the advanced properties into the scoring functions. The optimized retrieval function results will increase according to adding query terms to documents. Also, the increase in score is defined not proportional to the increase of query term into documents. Like the document 5470, the original Pivoted Length Normalization will just simply normalize its length and do the relevance calculation, since it is greater than the average document length. While it contains more query terms, such as complex, variable and etc, so optimized PN will increase score for this document ranking and put this document onto higher ranking position. That can be the reason behind optimized PN performs better than PN in the AP evaluation.