Project 2 Report

Junrong Yan

1. Expectation

The scoring functions perform as:
Okapi TF < Okapi TF-IDF < Okapi BM25
Okapi Laplace Smoothing < Okapi Jelinek-Mercer Smoothing

I expect that Okapi TF-IDF will get higher score than Okapi TF, because the Okapi TF estimates the importance of a term by the frequency of the term occurring in each document, that is, tf(d,i). However, the Okapi TF-IDF estimates the importance of a term by the frequency of the term occurring in all documents, that is log(D/df(i))

Based on Wikipedia, BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document. I think it be de different based on the length of document. Thus, I couldn't say which one is better, though BM25 may be better than Okapi TF-IDF, as BM25 is a method for scoring documents with relation to a query.

The Jelinek-Mercer Smoothing will be better than Laplace Smoothing. It uses formula $Pi = \lambda p + (1- \lambda)Q$ by training model on training set, which is based on the experience. And it will be better than Okapi TF, Okapi TF-IDF

Laplace Smoothing is also called add-one smoothing for solving the problem with 0 probability. Compare with all above scoring functions, it will be less effective because it gives too much probability mass to term which does not exist.

2 Analysis and evaluation of results

Below is the statistics of the results after evaluation.

Table1: GAP of each query and the avg

Query	TF	TF-IDF	BM25	Laplace	JM
202	1.0	1.0	0.007936507936	0.00552486187845	1.0
214	0.479253189668	0.513721109406	0.552663895383	0.494821940018	0.519476974132
216	0.429571309425	0.420159153778	0.486108146668	0.430052083989	0.469763570455
221	0.324451329029	0.319476339322	0.353742283232	0.396189252584	0.318542405494
227	0.111245126734	0.118265313381	0.238075307203	0.108598312914	0.188756485635
230	0.224963364539	0.192170737879	0.22457222983	0.256413304736	0.213190458956
234	0.573788681768	0.580525893211	0.720349687192	0.564840745318	0.684010775317
243	0.303562006188	0.302842076066	0.407035464478	0.225055269718	0.423413757681
246	0.0815355102738	0.0765041700636	0.12842731164	0.0719752526161	0.105939246518
250	0.360073005897	0.390974037315	0.139196715779	0.0429881697445	0.382930889029
avg	0.388844352352	0.391463883042	0.325810754934	0.259645919352	0.430602456322

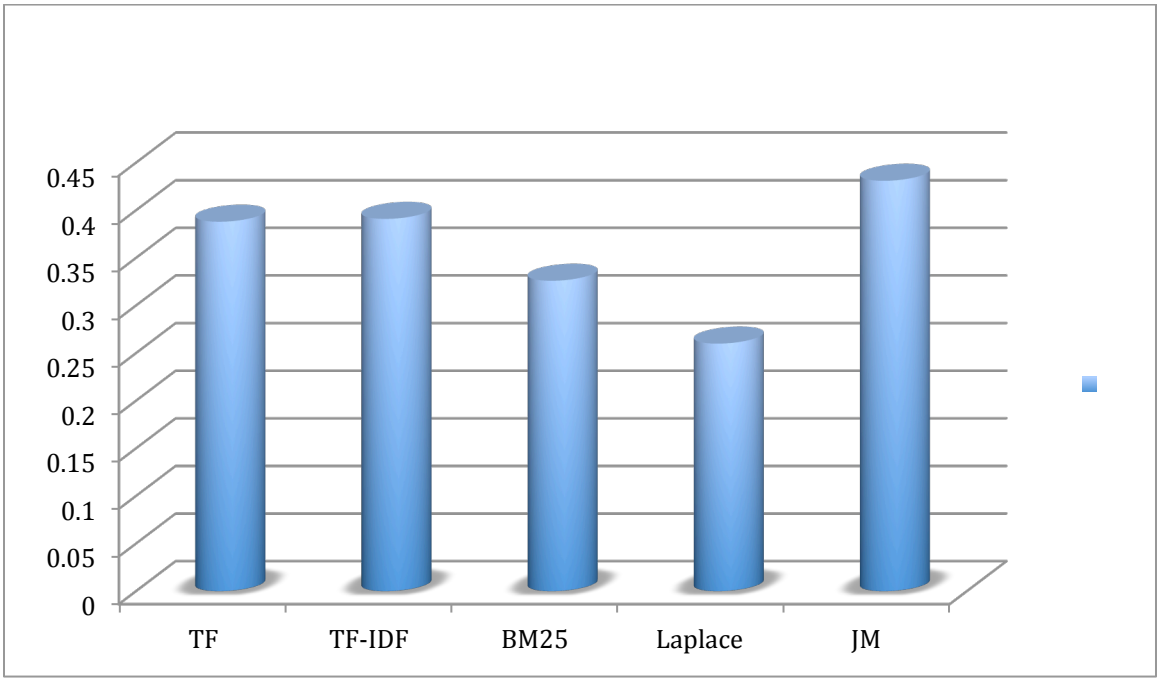

Chart 1: Graded Average Precision(GAP)

Based on table 1 and chart 1, we can see that Jelinek-Mercer Smoothing (JM) got the highest precisions in GAP. According to the formula of JM, in which λ can be [0,1], we can play with different λ value to see its effect. JM Smoothing is doing interpolation in linear, which is based on the experience, thus, it should be relevantly good. However, the parameter is not easy to estimate.

.

Okapi TF is using vector space model to estimate the importance of term by the term frequency in each document. In addition, it assumes that terms are independent with each other, which couldn't be true in real world. Thus, the graded average precision of this module would not be high. But this algorithm is simple framework for computing the rank.

Okapi TF-IDF get higher score than Okapi TF, but less than Jelinek-Mercer. The Okapi TF-IDF is a classical information retrieval term weighing model, which estimate the importance of a term in a document based on the formula : oktf(d,i) *log(D/df), which will be more accurate.

The language model with Laplace Smoothing got the lowest score among these five modules. The problem is that it adds

one to all frequency count and it assumes that each document has at least one term, even the documents is empty. Thus it gives too much probability mass to unseen n-grams, which leads to less precision. Based on the formula: $P_d(i)$= (tf +1)/(len(d) + V).

The BM25 get higher graded average precision score than Laplace. According to the lecture of this chapter, it ranks documents according to their relevance to a given search query based on binary independence model. In the formula of BM25, *k1, k2* and *K* are parameters whose values are set empirically. And b is used to reduce the influence to weight by the length of document. Usually, the term will occur in query only once, thus the formula (1+k2)*tf/(k2+tf) will not effect the precision. However, the length of document will be an important element to effect the precision.
Based on my index, the GAP of BM25 is smaller than TF and TF-IDF will be possible, since the length of document.

3 Extra : better-performing queries propose

Module 1: Okapi-TF
Offering the topic.xml, in which the terms of query should be more independent with each other, avoiding the phrase, and

relevant words, such as “white dog”, “green tea”, which will be very possible occurring in documents together.

Module 4: Laplace Smoothing

Offering the topic.xml, in which the terms of query should be more common, and occur in as most as documents.

Because, the Laplace Smoothing add I to all frequency counts, leading to much probability mass to unseen n-grams. Thus, using formula $P_d(i) = tf/(len(d)+V)$, when tf=0, set $\log(P_d(i))=0$. This will reduce probability mass.

Module 5: Jelinek-Mercer Smoothing

JM uses a λ of the weight attached to the document probability, query likelihood. Based on the formula, $Pi = \lambda p + (1- \lambda)Q$, to set λ based on splitting data into training and testing. When set different λ, to test the GAP of JM to improve better performing for queries.