Ranking functions: BM25 and variants BM25+ and BM25F

CS410
Michael Bernardoni (mlb12)
11/15/2020

Okapi BM25 [hereafter BM25 – Okapi refers to Okapi information retrieval system, where BM25 was first implemented at London's City University during the 1980s and 1990s] is a successful term frequency – inverse document frequency [TF-IDF] retrieval algorithm [1][2]. Variations of BM25 are currently state of the art TF-IDF algorithms used in text retrieval. This paper will discuss two new variants of BM25 that are extending the commonly applied BM25 algorithm: BM25+ [3] that addresses one of the weaknesses of BM25, the algorithm's ability to over penalize very large documents or over reward very short documents; and BM25F [4] that extends BM25 to take advantage of documents that have some structure to them (for example, headers, anchor text, main body, etc.) [4].

BM25

Okapi BM25 was first implemented at London's City University during the 1980s and 1990s, based upon research stretching back to the 1960s, and is a successful term frequency – inverse document frequency [TF-IDF] retrieval algorithm [1][2]. The history and derivation of BM25 is not the thrust of this paper but the cited references are an interesting read.

BM25 has several variants. One of the most common variant implements the following function:

f(q, d) = $\Sigma_{w \epsilon q \cap d}\ c(w,q)\frac{(k+1)c(w,d)}{c(w,d)+k(1-b+b\frac{|d|}{avdl})}\log\frac{M+1}{df(w)}$ [3 table 1]

Notation:

function *f* of query *q* and document *d* = sum over the unique words in *q, d*

parameters are *k* (0, +∞) and *b*(0 1)

c(w, q) count of word *w* in query *q*

c(w, d) count of word *w* in doc *d*

|d| length of document *d*

avdl average doc length

M total docs in collection, *df(w)* number of documents that contain word *w*

As can be seen, the formula contains 3 sections multiplied together:

Section 1 – query term frequency: *c(w, q)* is the count of word *w* in query *q*. Interestingly, the first implementations of BM25 had a parameter k3 to normalize the words in the query, similar to k in the formula and discussed below. It was discovered that in practice, however, that normalizing the term frequency for words in the query did not improve the algorithm. [4 section 3.5.1]

Section 3 – inverse document frequency: $\log\frac{M+1}{df(w)}$ is the inverse document frequency IDF normalization. In practice, a number of IDF functions can be plugged into the BM25 formula. Another common IDF algorithm is: $\ln(\frac{M-df(w)+0.5}{df(w)+0.5}+1)$ [1]. A discussion of the IDF component of the algorithm is not the intent of this paper.

Section2 document term frequency with document length normalization: $\frac{(k+1)c(w,d)}{c(w,d)+k(1-b+b\frac{|d|}{avdl})}$ is the term frequency TF section of the algorithm and is the topic of this paper. The TF normalization is controlled by 2 parameters: *k* (0, +∞), *b*(0 1).

The *b* parameter controls the document length nomination. As you can see from the formula, if *b* is set to 0, no normalization occurs. If *b* is > 0, the effect is to reward shorter documents and penalize larger documents – the result of $\frac{|d|}{avdl}$ gets larger as the document gets larger, however this result is in the denominator of the equation so larger documents are penalized and shorter documents are rewarded, pivoting around the average document length. The greater the b, the greater the reward/penalization.

The *k* parameter controls the sub-linear transformation of the term frequency. The interesting thing about this equation is that the transformation has an upper bound of *k*+1. Looking at the boundaries, when k = 0, the equations is bounded by 1 (k + 1). The transformation becomes a binary calculation, if the term exists, 1 if the term does not exist 0. On the other extreme, as *k* approaches ∞, the transformation approaches linear with no transformation.

Adding an upper bound to the equation is important. The upper bound protects against "spammers" who might introduce a document with a key word repeated multiple times in attempt to bias the retrieval algorithm.

Note: some implementations do not include the (*k* + 1) in the numerator [4 section 3.5.1]. As this is a constant, it does not affect the rankings, so it is proper to add this to the equation.

BM25+

While the *k* parameter establishes an upper bound for the term frequency, the document length nominalization controlled by *b* in the common B25 algorithm discussed above does not establish bounds. With the BMI formula $\frac{(k+1)c(w,d)}{c(w,d)+k(1-b+b\frac{|d|}{avdl})}$ it is easy to see that if the parameters *k* or *b* are set to large values, with large documents the $\frac{|d|}{avdl}$ in the denominator would have a large influence and as the document got larger the result would approach 0, and as such, a shorter document without all the query terms could possibly be scored similarly to a very long document that does contain all the query terms. It is intuitive to say, that in these circumstances, very long documents are overly penalized [3 section 3.1.1].

Interesting to note, very short documents can also be overly rewarded. Using the analysis above, it can be demonstrated that a very small document, exiting of query terms, would score quite high, even without all the query terms. Indeed, one can image if the query itself was in the document collection, the query document would score quite high due to the very short nature of the document (no extra words) along with all the query terms. However delivering the query back to the user would not be very useful.

BM25+ tries to address this deficiency.

BM25+ introduced 2 new constraints: (1) very long documents that contain query terms must not score close to, or below, short documents with zero query terms; and (2) short documents that only contain a small subset of query terms should not outscore a larger document with many query terms [3 section 4].

In order to satisfy these constraints, BM25+ introduces a new parameter δ into the equation $\frac{(k+1)c(w,d)}{c(w,d)+k\left(1-b+b\frac{|d|}{avdl}\right)} + \delta$. Clearly if δ is set sufficiently large, BM 25+ satisfies the constraints. The proposing paper proves that the constraints are satisfied when $\delta \geq \frac{k}{k+2}$ [3]. Experimentally, BM25+ works well when δ = 1 if training is unavailable [3 section 6.1].

Thus, by including one simple parameter, often set to 1, into the BM algorithm, BM+ solves the unbounded document length issues inherent with BM25.

BM25F:

The BM25 algorithm works on a document as if it is a single body of text. However, it is common for documents to have different segments, such as title, abstract, body for scientific articles (the cited article uses the term "streams" or "fields" this paper will refer to them as segments) giving the document some structure. BM25F utilizes this knowledge to improve the BM25 algorithm. The idea is that some segments may convey more relevancy that others [4 section 3.6].

One logical solution would be to evaluate these segments separately, using a scoring function like BM25, and then combine the scores with appropriate segment weights. That is, one would segregate on the segment, then analyze utilizing the terms in that segment, and in the end adjust using the weights.

When one analyses this approach, one can conclude that the term frequency normalization is compromised as a word appearing once in each segment will get the benefit of the "first word" score on all 3 words, one in each segment. BM25F argues for a different approach [4 section 3.6.1].

The core of the BM25F algorithm is to first weight the term frequency across the entire document, adjusted by segment weights. The algorithm then uses these term weighted scores (one for each term) to score the final document [4 section 3.6.3]. BM25F uses the following notations:

Segments (or streams) s = 1 to S
Segment length sl_s
Segment weight: v_s
tf_{si} = term frequency of term i in segment s

This can be described as a vector of vectors; each document has a vector of segments, and each segment has vector of terms.

The first step of BM25F is to calculate a weighted score for each term by summing (for each segment) the segment weight multiplied by the term frequency for that term in that segment:
weighted term frequency $wtf_i = \sum_{s=1}^{S} v_s \ tf_{si}$

Similarly, you calculate a weighted variant of the total document length:
weighted document length $wdl = \sum_{s=1}^{S} v_s \ sl_s$

BM25F then plugs the weighted term frequencies and the weighted document length into the BM25 algorithm.

$\frac{(k+1)wtf_i}{wtf_i+k(1-b+b\frac{wdl}{avdl})}$ [4 section 3.6.3].

BM25F is an easy computable extension of BM25, and it allows the retrieval algorithm to utilize information contained in the structure of the document, even documents with simple structures.

One may note that the BM25F formula discussed above contains only 1 set parameters *b* and *k*. Thus, this form of BM25F is noted as the simple form. The article as a more complex form that allows the parameters b and k to be unique per segment. Please refer to the cited reference for more information [4 Section3.6.3].

Conclusion:

Variations of BM25 are currently state of the art TF-IDF algorithms used in text retrieval. This paper discussed two new variants of BM25 that are extending the commonly applied BM25 algorithm: BM25+ and BMF. BM25+ improves the algorithm by adding one simple parameter, commonly set to 1, to resolve the bounds issue of the document length normalization contained in BM25. BM25F extends BM25 to take advantage of documents that have some structure to them (for example, headers, anchor text, main body, etc.). Both BM25+ and BM25F are easily computable.

Citations:

1. *Stephen E. Robertson; Steve Walker; Susan Jones; Micheline Hancock-Beaulieu & Mike Gatford (November 1994). Okapi at TREC-3. Proceedings of the Third Text REtrieval Conference (TREC 1994). Gaithersburg, USA.*
2. *Stephen E. Robertson; Steve Walker & Micheline Hancock-Beaulieu (November 1998). Okapi at TREC-7. Proceedings of the Seventh Text REtrieval Conference. Gaithersburg, USA.*
3. Yuanhua Lv and ChengXiang Zhai. *Lower-bounding term frequency normalization.* In Proceedings of CIKM'2011, pages 7-16.
4. Stephen Robertson & Hugo Zaragoza (2009). "The Probabilistic Relevance Framework: BM25 and Beyond". *Foundations and Trends in Information Retrieval*. **3** (4): 333–389. CiteSeerX 10.1.1.156.5282. doi:10.1561/1500000019.