Scribes: N. Sadat Shami, Gilly Leshed

Outline

1. An empirical approach to information retrieval (IR)
2. Reminder of the Vector Space Model (VSM)
3. Issues with document length normalization

1. An Empirical Approach to Information Retrieval (IR)

Many findings in IR were developed through empirical work, rather than as a result of theory development. For example, the Vector Space Model (VSM) describes a way that a collection of documents can be represented for information retrieval, but there was initially no theoretical underpinning for this model other than some intuitions. The domain of traditional IR is guided by striving to improve and optimize performance (specifically precision) over previous work. The mere demonstration of satisfactory performance results is sufficient to accept one's work and apply it into a working system.

2. Vector Space Model (VSM) Fundamentals

Recall from the previous lecture that the VSM is used to represent documents from a corpus as vectors in the following manner:

Document d is represented as vector $\vec{d} = [d_1,\ldots,d_j,\ldots,d_m]^T$, where d_j represents the weight of term $v^{(j)}$ in document d, and there are m terms in the vocabulary.

It is generally accepted that the following three concepts should be incorporated into the term weights of a document: idf_j, $tf_j(d)$, and $norm_j(d)$.

a. idf_j

idf_j (inverse document frequency) is an attribute of a term in a corpus, representing the rareness of the term. Measures of this quantity decrease with the fraction of documents in the corpus that contain the term $v^{(j)}$. A high value of idf_j indicates that the term is rare, and therefore more significant for distinguishing between documents. The lower the measure is, the more widespread the term is in the corpus and across documents, decreasing the term's distinctive power between documents.

For example, in a computer science document collection, the term "computer" may appear in many documents. It will therefore have low idf_j value, since it does not assist in discriminating between the documents. On the other hand, if the term "parallel processing" appears in only some of the documents in this collection, then this term would have a higher idf_j value, assisting in distinguishing the subset of documents that discuss the topic of parallel processing.

Note that idf_j is dependent on the entire corpus and not on a specific document.

b. $tf_j(d)$

$tf_j(d)$ (term frequency) represents the occurrence of term $v^{(j)}$ in document d, increasing with the frequency of the term in the document. A high value of $tf_j(d)$ implies that the term is representative of the document. For instance, if a document contains the word "cat" many times, there is good chance that the document is about cats.

c. norm$_j$(d)

The raw number of terms in a document creates a bias against the short documents, as long documents simply have more terms in them. Another motivation for normalization is spam: documents that "hide" certain terms to increase the term frequency within them. Therefore, a normalization function is required. This concept is expanded in section 3 below.

d. Document Relevance to Query

The rationale behind the VSM model for IR is that both queries and documents are represented in a vector space. The higher the similarity between the query and the document, the higher the document is ranked as relevant for the query.

$$\vec{q} \cdot \vec{d} = \sum_j q_j \times \frac{tf_j(d) \times idf_j}{norm_j(d)} = \sum_j \underbrace{(q_j \times idf_j)}_{\text{Same for all documents}} \times \underbrace{\frac{tf_j(d)}{norm_j(d)}}_{\text{Specific for every document}}$$

In this representation, a document will have higher chances of being relevant for a query if it contains a distinctive query term (i.e., high idf$_j$) that occurs many times (i.e. high tf$_j$(d)), although the normalization factor plays a role in potentially altering this behavior.

3. Document Length Normalization Issues

This section discusses a SIGIR '96 paper by Singhal, Buckley, & Mitra: "Pivoted document length normalization".

a. Why Normalize Document Length

There are two main reasons why term weights should be normalized, reducing the advantage that long documents have just because they have more terms in them:

1. Higher term frequencies: long documents tend to repeat the same term over and over. As a result, the term frequency of long documents tends to be higher, increasing the average contribution of its terms towards the query-document similarity [1].
2. More terms: long documents are likely to have a greater variety of terms than short documents. This increases the number of non-zero term frequency values for the long documents, increasing the number of matches between a query and a long document, and increasing the query-document similarity [1].

b. Normalization Functions

Various normalization approaches can be used. For example, one common normalization function is the **maximum term frequency**: $norm(d) = \max_j tf_j(d)$. This function attacks the first reason for normalization of higher tf values due to repetitions, but does not deal with the second reason of many non-zero tf values. Therefore, this function tends to favor the retrieval of longer documents. This approach can also be problematic from the following reasons:

1. There could be outliers, i.e., a document with a very-very large number of occurances of a specific term, not representative of the rest of the document.
2. A document in which the most frequent term appears X times and the rest of the terms appear fewer than X times should not be treated the same as another document in which many terms appear X times.

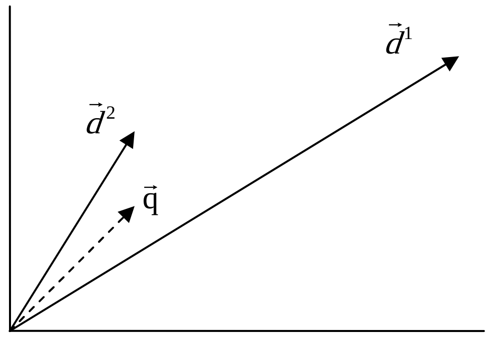
Another widely used normalization function is **cosine normalization**, and is computed by:

$$\mathrm{norm}_j(d) = \underbrace{\mathrm{norm}(d)}_{\text{“j” is removed as the value is computed by summing over all terms}} = \sqrt{\sum_j \mathrm{tf}_j(d)^2}$$

Using the cosine normalization function, the normalization value is higher as due to both higher term frequencies (reason 1 above) and more non-zero tf-s (reason 2). As a result, the following effect on the query function is observed:

$$\text{Score function: } \vec{q}\cdot\vec{d} = \underbrace{\|\vec{q}\|_2}_{\text{Same for all documents}} \underbrace{\|\vec{d}\|_2}_{=1} \cos(\angle(\vec{q},\vec{d})) \overset{\text{rank}}{=} \cos(\angle(\vec{q},\vec{d}))$$

The rank function is therefore only dependent on the directionality in the vector space, and the length of the document vector is ignored. For example:



In this example, the angles between the query vector and the two document vectors are equal, so that:

$$\cos(\angle(\vec{q},\vec{d}^1)) = \cos(\angle(\vec{q},\vec{d}^2))$$

But the document vector lengths are different.

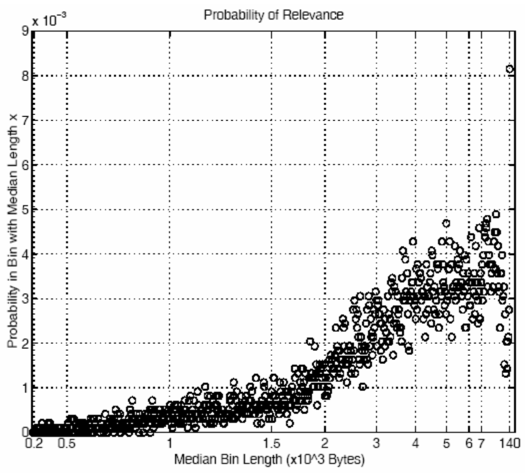
c. The Relevance/Retrieval Gap

Using normalization functions that ignore document length is appropriate if length was not correlated with relevance to the query. This matter was examined by Singhal et al.

Using the TREC system with 50 queries and their given relevant document[1], for each document length (when the documents were segregated into 1000 document bins), the number of relevant documents of this length was counted. This count was then divided by the number of relevant documents, indicating the probability of a document to be of a certain length given it is relevant:

$$\frac{\#(d \in \mathrm{Rel}(\mathrm{length}))}{|\mathrm{Rel}|} = P(d \in \mathrm{bin}(\mathrm{length}) \mid d \text{ is relevant})$$

This value was then plotted against the length of the document. Singhal et al found that document vector length and relevance to query are in fact correlated:



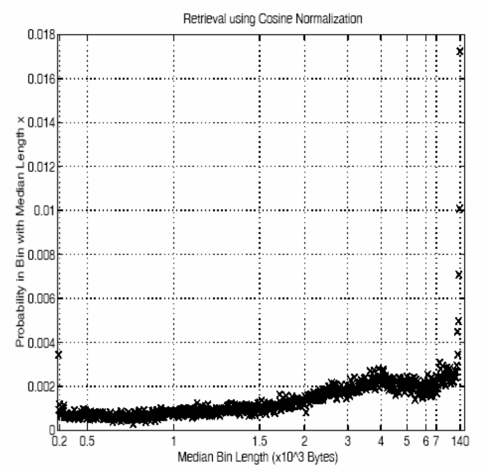
Note that if no correlation exists between document length and relevance, then the distribution of query relevance should be uniform, not increasing with the length of the document. The results reveal that relevant documents are more likely to be longer (but not necessarily the other way around). The results are not surprising. Look at the second reason for normalization discussed above: longer documents simply have more different terms in them, suggesting that they discuss more topics in them. As a result, longer documents have higher chances at "hitting" a certain query.

Similarly, Singhal et al plotted the results of retrieving documents using the cosine normalization function: for each document length, the number of retrieved documents of this length was counted, and then divided by the total number of retrieved documents. This generates the probability of a retrieved document to be of a certain length:

$$\frac{\#(d \in \mathrm{Ret}(\mathrm{length}))}{|\mathrm{Ret}|} = P(d \in \mathrm{bin}(\mathrm{length}) \mid d \text{ is retrieved})$$

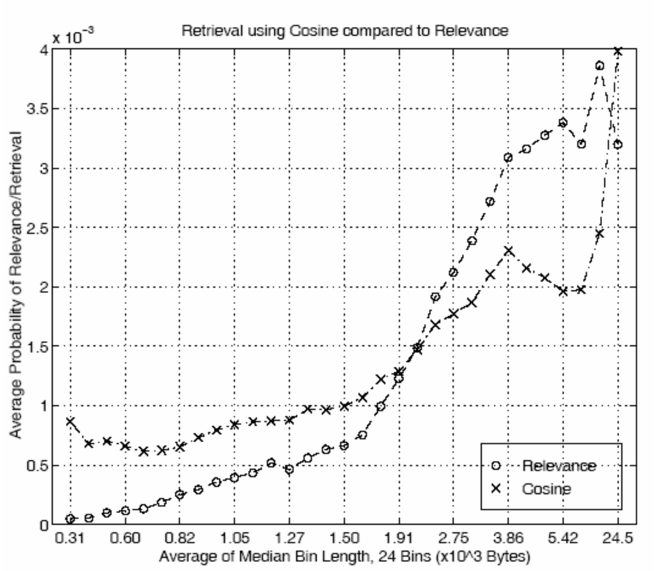
[1] The relevant documents provided to each query in TREC were not selected by hand, as the corpus consists of on the order of hundreds of thousands of documents. Rather, they were derived by manually checking each document that appears in several IR systems. This creates a sampling bias toward these systems, and of course, the recall rate is unknown: we do not have access to the relevant documents that did not come up from the systems. This issue is discussed in the next lecture.

Plotting the value against the document length revealed that retrieval that uses the cosine normalization function is positively correlated with length:

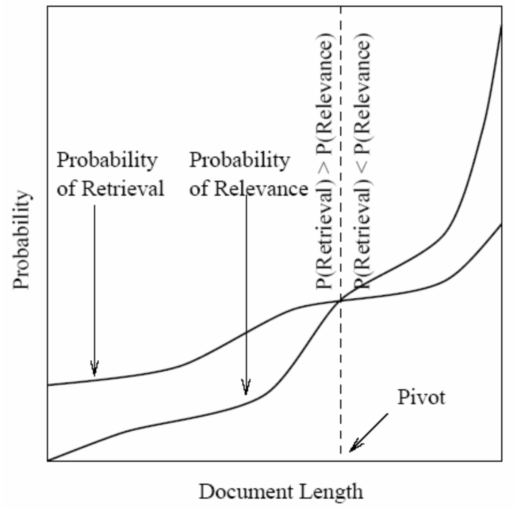


On one hand, the result is surprising: recall that the cosine normalization function, $\text{norm}(d) = \sqrt{\sum_j \text{tf}_j(d)^2}$, is such that long documents will have higher normalization values, cutting down the score function $\vec{q} \cdot \vec{d} = \sum_j (q_j \times \text{idf}_j) \times \frac{\text{tf}_j(d)}{\text{norm}_j(d)}$. On the other hand, we now know that length is correlated with relevance, and that using cosine normalization yields satisfactory retrieval results, and so the positive correlation of retrieval and length seems reasonable.

When the two plots for relevance and retrieval where smoothed and plotted against each other on the same graph, the following result was obtained:



This indicates that cosine normalization tends to favor shorter documents: retrieval based on cosine normalization tends to retrieve short documents with a higher probability than their probability of relevance, and to retrieve long documents in a lower probability than the probability of their relevance. So, although the cosine normalization favors long documents over short documents, its bias against short documents and toward long documents is not as strong as the relevance curve suggests it should be. This idea is illustrated in the following schematic figure:



Note: Pivot in the above diagram was later renamed 'crossing point' in following classes

d. Next Lecture

To fix the gap between the relevance and retrieval curves, Singhal et al suggested conceptually tilting the retrieval curve around the pivot: the point at which two curves intersect (in practice, they tilted the norm function, not the probability function). They proposed to fix the retrieval curve by applying a linear transformation on the original normalization function:
norm'(d) = m (norm(d)) + b

The next lecture will complete covering Singhal et al's pivoted document length normalization that they presented in the paper.

References

[1] Singhal, A., Buckley, C., and Mitra, M. (1996). Pivoted document length normalization. SIGIR '96.

Finger Exercise

In thinking about cosine normalization (or any normalization really!), it is important to consider how a user formulates a query, since much of the similarity method depends on the query term(s). Let's say a user is using a search engine to gather information to buy a car. Different users employ different query terms while attempting to formulate their search queries. Let's say a particular user has entered in the words 'automobile manufacturer' (without the quotes) into her favorite search engine. Google returned the followed results in rank order (retrieved 6:40 AM, Monday February 6, 2006):

1. http://www.autoalliance.org/index.php?flash=yes (*d1*)
2. http://www.jama.org/ (*d2*)
3. http://www.aama.com/ (*d3*)[2]
4. http://www.naamsa.co.za/ (*d4*)
5. http://www.indexoftheweb.com/Automobile/Manufacturers.htm (*d5*)

For convenience, assume:
- 'automobile' occurs in 90 of the documents in the collection
- 'manufacturer' occurs in 55 of the documents in the collection
- the number of documents in the collection, N, is 500

a. Rank these 5 documents by computing the similarity of the query 'automobile manufacturer' with each of the documents *d1-d5* using the cosine normalization function. As you may recall, the similarity measure between a document and a query, is given by the formula:

$$sim(d,q) = \sum_j q_j \cdot \frac{tf_j(d) \cdot idf_j}{norm_j(d)}$$

Where
tf_j(d) = frequency of term j in document d
q_j = frequency of term j in query q
idf_j = inverse document frequency of term j in collection C

The cosine normalization function is:

$$norm_j(d) = norm(d) = \sqrt{\sum_j \left(tf_j(d)\right)^2}$$

Use $\mathrm{idf}_j = \log\left(\frac{\mathrm{N}}{\mathrm{n}_j}\right)$ for the inverse document frequency calculation.

Where
n_j = number of documents in C that contain term j
N = number of documents in C

[2] *In d3, you will notice a redirect from www.aama.com to www.amalabs.com. It appears that the American Automobile Manufacturer's Association (AAMA) has disbanded and a nailcare salon bought the domain. This accounts for a) some of the empirical anomalies we will find in different ranking schemes, and b) a good example of the dangers of web indexing.*

b. Read the contents of these 5 documents and produce your own ranking of them based on how well each of them satisfies a user's information need of buying a car.
c. Compare the rankings produced by the search engine, the similarity method and your own ranking. Discuss any discrepancies you find. What additional information about the documents (other than the number of occurrences of each query word) did you use to rank them which your favorite search engine and the similarity method did not take into account?
d. Describe an algorithm that would enhance the similarity method in a way that reflects your own ranking better.

Solution

a. *Similarity values between query q and documents d1-d5:*

Notation:
$tf_1(i)$ = frequency of term "automobile" in document i
$tf_2(i)$ = frequency of term "manufacturer" in document i
n_1 = number of documents in the total collection that contain term "automobile"
n_2 = number of documents in the total collection that contain term "manufacturer"

The following shows the different stages in the calculation:

Stage 1: Calculate the normalization measures
Using MS-Excel, the contents of the webpages were copied from the webpage, and using simple counting and sorting Excel functions the term frequencies for all terms in the webpage were calculated. This is the text in the body of the webpage and excludes terms in the title bar and on graphics. As an example, the 15 most frequent terms in *d5* (the longest document out of the five), which contains 802 unique terms is provided below:

term	**count**
the	77
and	71
our	54
to	51
of	42
in	31
a	26
you	21
information	19
for	18
is	18
vehicles	18
site	17
welcome	16
page	14

From this calculation, the term frequencies of the query terms were found, and the normalization values were calculated by $\sqrt{\sum_j \left(tf_j(d)\right)^2}$.

The following table provides term frequencies for 'automobile' and 'manufacturer' as well as norm(d) for documents *d1-d5*. Note that stemming was not applied in the calculations as well as dealing with synonyms, leading to zero counts for some of the term frequencies.

	tf_1(i)	**tf_2(i)**	**norm(d)**
d1	1	0	8.831760866
d2	3	0	39.47150871
d3	0	0	65.88626564
d4	1	1	55.09083408
d5	7	2	167.6633532

Stage 2: calculate the inverse document frequency
Given the assumptions above, n0 = 90, n1=55, and N=500:

$$idf_1 = \log(N/n1) = \log(500/90) = 0.7447$$
$$idf_2 = \log(N/n2) = \log(500/55) = 0.9586$$

Stage 3: for each document calculate the weight = $\frac{\mathrm{tf}_{\mathrm{j}}(\mathrm{d}) \cdot \mathrm{idf}_{\mathrm{j}}}{\mathrm{norm}_{\mathrm{j}}(\mathrm{d})}$ for each query term, and sum over the terms. This provides the similarity of the query and each document. The weights of the query terms do not affect the document scores, as both terms appear once in the query.

	automobile	**manufacturer**	**sum**
d1	0.084323784	0	0.084323784
d2	0.056602409	0	0.056602409
d3	0	0	0
d4	0.013518174	0.023364904	0.036883078
d5	0.031092617	0.015354483	0.0464471

Ranking based on similarity:

Rank	Document
1	**d1**
2	**d2**
3	**d5**
4	**d4**
5	**d3**

b. *Personal ranking*

Rank	Document
1	**d5**
2	**d2**
3	**d1**
4	**d4**
5	**d3**

Ranking based on similarity

Rank	Document
1	**d1**
2	**d2**
3	**d5**
4	**d4**
5	**d3**

c. This example clearly illustrates the problems inherent in query formulation and subsequent retrieval of results by search engines. IR researchers have long recognized that it is quite hard for users to express their query using effective search terms. And based on user-entered terms, it is hard for search engines to return documents that a user will find relevant. In this example, there are differences between the search engine ranking, similarity method ranking, and our personal ranking. Between our personal ranking and the ranking based on similarity, we see that the only difference is that *d1* and *d5* are swapped. Indeed, it seems that *d5* is simply being over penalized for length by cosine normalization – a problem we will discuss and provide a fix for in a future class. Another glaring fact is that the document we found most relevant was ranked last by Google! The document ranked 3rd by Google has absolutely nothing to do with automobile manufacturers, although this is most likely because of old bookmarks still pointing to the AAMA site.

As humans, we could easily understand the context of the documents just by glancing at them, and decide which one is most relevant to our query. Perhaps this could be explained better through Information Foraging theory by Pirolli & Card [3]. According to this theory, individuals select where to go in the information environment based on estimates of the value they are likely to get by taking prospective routes. An "Information scent" signals to the human user the merit behind a piece of information, and is used in the estimation process. However, IR algorithms do not necessarily rely on this understanding and therefore do not have this capability. The majority of documents returned by our query are about automobile manufacturers associations, which do not give us much information about an automobile's manufacturer where we could potentially obtain information about purchasing an automobile. With certain queries, search engines still need to understand how humans formulate queries and return results accordingly. As much as different documents will be considered most relevant by different users based on the same query, this remains a hard information retrieval problem.

d. Relevance feedback is a technique that comes to mind to improve the similarity measure [1]. Relevance feedback is motivated by the fact that it is easy for users to judge some documents as relevant or non-relevant for their query. Using such relevance judgments, a system can then automatically generate a better query (e.g. by adding related new terms) for further searching. In general, the user is asked to judge the relevance of the top few documents retrieved by the system. Based on these judgments, the system modifies the query and issues the new query for finding more relevant documents from the collection.

Another approach could be based on combining query terms with link structure in web documents as described by Amento et al. [2]. But it should be noted that this is of course, precisely what Google is doing (among other things). Their method involves obtaining a seed set of pages that refer to the query term(s) using traditional IR techniques such as *tf x idf*. They then use in-degree calculations to determine the most relevant documents. The document with highest in-degree from the seed pages would be the most relevant.

References

[1] I. Ruthven, and M. Lalmas. A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review*, 18(1) (2003)

[2] B. Amento, L. Terveen, and W. Hill. Does "Authority" Mean Quality? Predicting Expert Quality Ratings of Web Documents. In *Proceedings of ACM SIGIR '00,* ACM: New York, 2000.

[3] P. Pirolli, and S. Card. Information Foraging. *Psychological Review*, 106(4), 643-675, 1999.