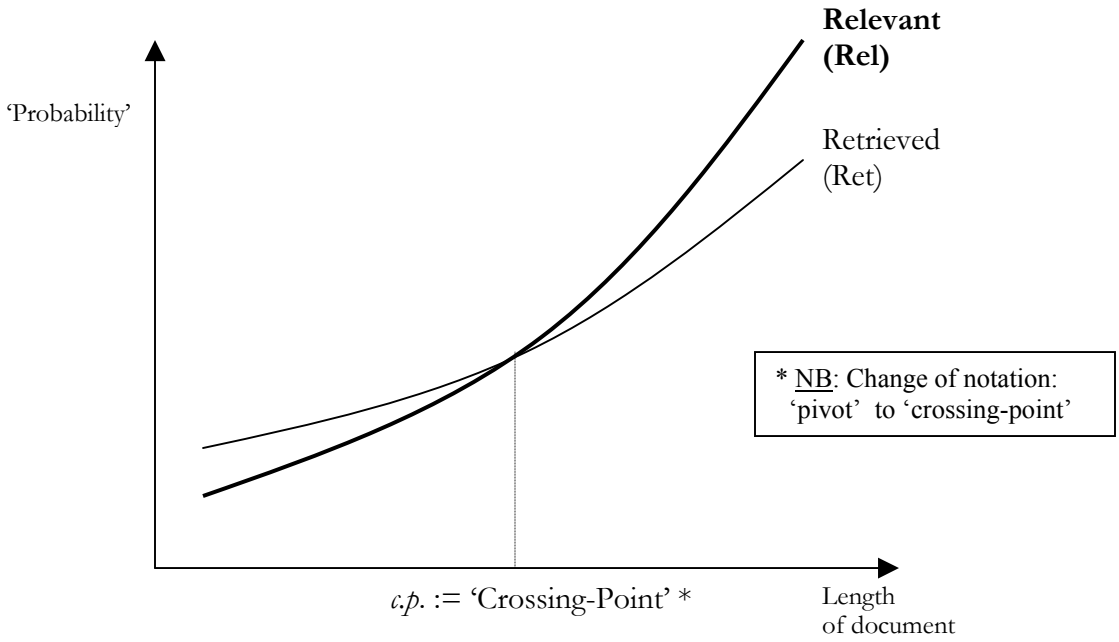
Representing and Accessing [Textual] Digital Information (COMS/INFO 630), Spring 2006
Lecture 3: 2/2/06 **Pivoted Length Normalization, Conclusion**
Lecturer: Lillian Lee
Scribes: Nicholas Hamatake (nh39), Min Park (mp257)

Recall from 1/31/06,



The Point of this graph: Rel and Ret curves do not match; however, their slopes are in the 'right' direction.

- Cosine normalization over-retrieves short documents
  (*short* with respect to the crossing-point)
- Curves are nonlinear; so please take care to keep track of which curve is described by which function. That is, we are going to compute a linear function of document norm, not of the retrieved or relevant functions plotted above.

Observations

1. *Curve construction: Smoothness and Reliability*
   Why are these curves so smooth? How reliable are the data points?
   Why don't the curves drop down to the x-axis?
   For instance, if there are no documents of lengths n for $c.p. - 5 < n < c.p.$, but many documents of length $c.p. - 5$ and $c.p.$?
   Also, if there are only a few documents of a given length, can we really trust the calculated probability values?

   Construction:
   1. Sort documents by length
   2. Divide into 1000-document bins
      let x = median length for a bin in the graphs

$\Rightarrow$ Each data point on the graph is based on the same (large) amount of data, a nice feature.
However, the last bin may have fewer than 1000 documents.

2. *Bias in Rel set*

Rel: the set of relevant docs derived by the 'pooling' of several systems' output
vs.
The full set of relevant documents in the entire corpus ("Truly relevant set")

Problem: There may be some bias in the *Rel* set. Although this set is hand checked by humans for relevance, this does not imply that *Rel* set is truly representative of the actual relevant set, since we don't know if the relevant documents in the pool represent an unbiased sample from the entire set of relevant documents.

Zobel, SIGIR '98 addresses this problem in a nice paper:

How can we estimate the number of truly relevant documents (estimating $|Rel_{true}|$)?

An idea proposed in class:

We could randomly sample documents from the entire corpus, and check them for relevance. Unfortunately, the expected number of relevant documents over a large general corpus is low, so this process is probably too time-consuming.

Zobel proposes a simple idea using the notion of 'arrival rate'

R_k := number of relevant documents in a pool of depth k
R_{k+1} := number of relevant documents in a pool of dept k+
} k the 'pool depth' parameter (ex. k = 100 is the setting in TREC)

Consider the aggregation of the top k documents returned by each system (repeats discarded),

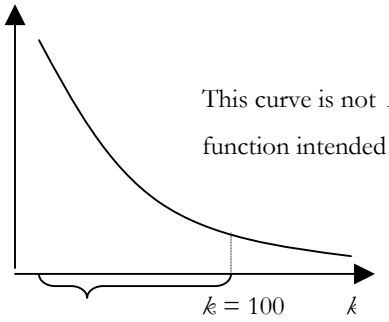
$$R_{k+1} - R_k \qquad (1.1)$$

We expect this quantity (1.1) to be decreasing in k. For instance, if all relevant documents have been captured in R_k, then the number of relevant documents captured in R_{k+1} should be equal to R_k and (1.1), their difference, should equal 0.

We have a function based on (1.1), and we have data; we can fit the function to the data. Note that the actual function needs free parameters in order to adjust the fit to the data.

Find parameters that fit the data for $k < 100$, then extrapolate for $k \geq 100$.

Result:



Exhibits good fit to the actual $R_{k+1} - R_k$ data.

If we believe that this curve is a good approximation to $R_{k+1} - R_k$, how could we estimate the size of the true relevant set? This would be the area under the curve!

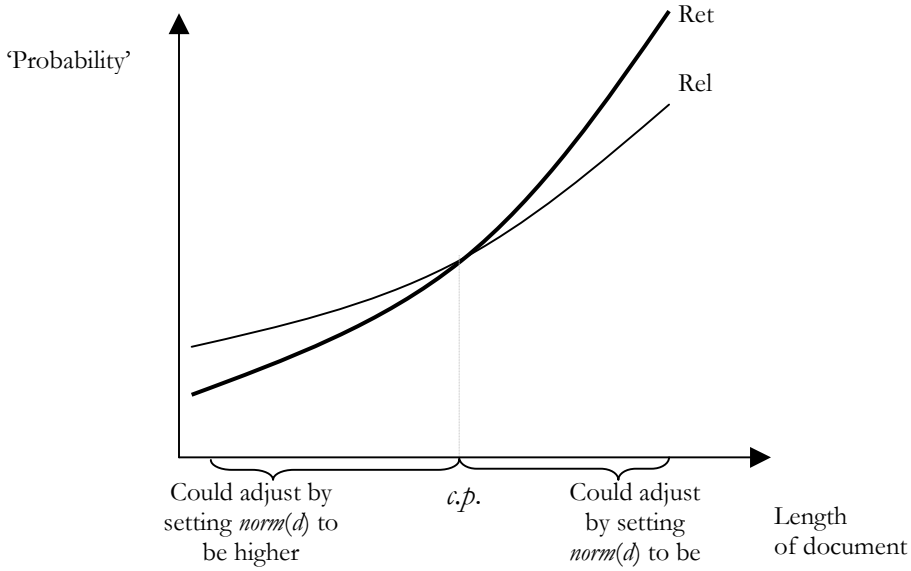
However, note that we expect that the extrapolated values will be less trustworthy for very large values of k, so we might not want to take the entire area under the curve. Instead we could cut it off at some point.

What Zobel did:

For k = 200, 500 (TREC 5), 6,707 and 9,358 were obtained as the respective areas under the curve, and hence (under)estimates for the number of relevant documents in the entire corpus. However, for k = 100, the number of relevant documents seen was equal to 5046. This implies that the proportion of missing in *Rel* is significant, and so **Recall is being underestimated.**

The Point: If the graph on page 1 credible, given that we believe that many relevant documents were unseen? Well, we still don't have evidence one way or the other that the sample of relevant documents is significantly biased in an important way.

Take this graph as given for now. What do we do? How do we 'fix' it so that the two curves are closer to matching?

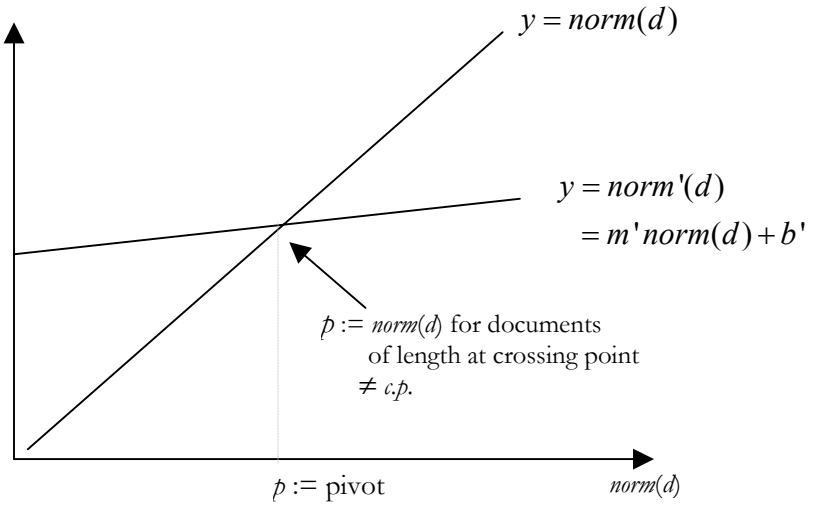


Linear Correction of *norm*(*d*)

$$norm'(d) = m'\,norm(d) + b' \tag{1.2}$$

Define p := pivot, $norm(d)$ for some d of length *c.p.*

> **Note!** Pivot, p is **not** equal to the Crossing-point, *c.p.*, which is a length
> The x-axes of the two graphs above are not the same!



Note that we would like for the new line to have a more gradual slope.
Unfortunately, this linear correction creates two new free parameters. How can we get rid of some these unknowns?

At $norm(d) = p$,

$$norm(d) = p$$
$$= m'p + b' = m'\,norm(d) + b'$$
$$\Rightarrow b' = p(1 - m')$$

Substitute into (1.2):

$$norm'(d) = m'\,norm(d) + p(1 - m') \tag{1.3}$$

We have parameters m' and p. Let us consider them as free parameters and think about the task as searching for values that yield good retrieval results (rather than directly choosing values that fit some relevance-labeled data). We want to make (1.3) cleaner, more elegant, and ease the search burden—we want to get rid of some (one) parameter.

Reducing a free parameter in equation (1.3)

Claim: $norm'(d) \overset{rank}{=} \alpha\, norm'(d)$

if α is a constant with respect to d

That is in ranking, $\sum_j q_j idf_j \frac{tf_j(d)}{norm'(d)} = \frac{1}{\alpha}\sum_j q_j idf_j \frac{tf_j(d)}{norm'(d)}$

Then consider $norm''(d) = \frac{1}{p(1-m')} norm'(d)$ (1.4)

$$= \frac{m'}{p(1-m')} norm(d) + 1$$

We will now 'get rid of' *p* as a search parameter by fixing it, then adjusting $\frac{m'}{(1-m')}$ appropriately. The idea is to pick a good *p,* and correct it later.

Try $p = \overline{norm}$,

= the average *norm*(*d*) over all *d*s. Note that it is easy to compute!

Claim: $\exists m''$ s.t. $\frac{m''}{(1-m'')\overline{norm}} = \frac{m'}{(1-m')p}$

Proof: Solve for m''.

Then let $norm'''(d) = \frac{m''}{(1-m'')\overline{norm}} norm(d) + 1$ (1.5)

We can search for m'' over some continuous range, which is equivalent to searching for the previous combination of parameters over some continuous range.

Now, to focus attention on the relation between $norm(d)$ and $\overline{norm}$, we multiply (1.5) by $(1-m'')$ (which doesn't affect ranking):

$$(1-m'')norm'''(d) \overset{rank}{=} m'' \frac{norm(d)}{\overline{norm}} + (1-m'') =: norm^{(iv)}(d) \quad (1.6)$$

How can we interpret this expression? Suppose for some *d*, $norm(d) = \overline{norm}$; then $norm^{(iv)}(d) = 1$.
A document with the average length in the corpus is of 'appropriate' length.
Aside: In 1994 Robertson and Walker derived this exact normalization in (1.6) using different motivation!

How did this linear correction do?
Relevant and Retrived curves were much closer and there were 6~12% improvements in average precision results.

Could they have overfit?
To check, they applied the method on six different copora; the optimal (m")'s were relatively stable (≈ 0.7)

To summarize…
Good research looking easy!
This research took a normalization function (cosine) as given, then fixed it in an empirically effective manner.
But we might ask, "why not try to come up with a better normalization function in the first place?"

…Next class.

Questions:

1) Suppose we have a corpus in which $\overline{norm} = 21$ (under classic cosine normalization). Consider a document d with a normalization factor $norm(d) = 6$ (again, under standard cosine normalization). Apply pivoted cosine normalization with a slope of $m = 0.7$ (the primes are hereafter omitted for clarity) to answer the following:

   a) What is $norm'(d)$, the pivoted normalization factor of document d?

   b) If $\vec{d}$ is the vector representation of document d using cosine normalization and $\vec{d}'$ is the vector representation of document d using pivoted cosine normalization, how does $\left\|\vec{d}\right\|_2$ compare to $\left\|\vec{d}'\right\|_2$?

   c) By re-normalizing, we are simply multiplying each document vector by a scalar. Why does this affect ranking, while other situations in which we multiply each document by a scalar do not?

2) What happens to the pivoted normalization factor as the slope *m* increases? Can we say anything about what *m* intuitively represents?

Solutions:

1a) 0.5

1b) $\left\|\vec{d}'\right\|_2$ is much larger (by a factor of 12).

1c) The scalar factor is document-dependant and thus can affect ranking. Only multiplication by a constant with respect to all documents is guaranteed to preserve rank.

2) As *m* increases, the ratio of document length to average document length becomes more heavily-weighted in the pivoted normalization factor. Intuitively, we can say that *m* represents the degree to which we value document length as a factor in determining document rank, i.e., "how much does length matter?" If we set *m* = 1.0, applying the pivoting process would just involve dividing each document norm by a constant (the average document norm), which we know has no effect on ranking; standard ranking is appropriate if we trust length.