CS 630 Notes: Lecture 5
Lecturer: Lillian Lee

Notes by Matt Connolly and Danish Mujeeb

February 9th, 2006

1 Review of Classic Probabilistic Retrieval Model

Previously we modeled the problem of retrieval as follows: we will try to calculate the probability $P(relevent|doc)$ given a fixed query q. We then proceeded with the following steps.

1. To make sense of the original proposition, we converted doc d to an attribute vector, where attributes are "kind of" based on terms:

$$doc\, d \rightarrow \vec{a}\,(d) = (a_1\,(d)\,, ..., a_m\,(d))$$

   Hence our probability model gets converted to $P\left(R = y|\vec{A} = \vec{a}\,(d)\right)$

2. We then performed a "Bayes' Flip" to rearrange and get rid of certain terms that were constant with respect to a given document.

3. We then did some factoring and ended up with the following expression:

$$\prod_j \frac{P\,(A_j = a_j\,(d)\,|R = y)}{P\,(A_j = a_j\,(d))}$$

   Steps 1 - 3 were motivated by estimation concerns. In the end, we will have to estimate the values of all of these terms anyway. Hence we built these concerns into our derivation, anticipating that these steps will make the estimation easier when we get to it.

4. We then reintroduced the query q into the model. The query is the only information we have about possible relevance, and adding it to our model will allow us to reduce the number of R-dependent terms. We proceed with the following assumptions:
   Assumption: A_j corresponds to $v^{(j)}$
   Assumption: If you have a term that is not in q, i.e. if $v^{(j)}$ not in q, then $P\,(A_j = a_j\,(d)\,|R = y) = P\,(A_j = a_j\,(d))$. Hence we get

$$\prod_{j:q_j=1} \frac{P\,(A_j = a_j\,(d)\,|R = y)}{P\,(A_j = a_j\,(d))}$$

   Now we are only dealing with terms in our query, so we have reduced some of the "noise" in our probability model.

2 Restricting to Terms Only in The Document

Here is a new idea: we will only concern ourselves with terms that occur in our document, i.e. $a_j(d) > 0$. We can separate our total probability into two probability products for the two cases $a_j(d) > 0$ and $a_j(d) = 0$, which gives us

$$\prod_{j:q_j=1,a_j(d)>0} \frac{P\left(A_j = a_j\left(d\right)|R=y\right)}{P\left(A_j = a_j\left(d\right)\right)} \times \prod_{j:q_j=1,a_j(d)=0} \frac{P\left(A_j = a_j\left(d\right)|R=y\right)}{P\left(A_j = a_j\left(d\right)\right)}$$

In order to get rid of our dependence on a particular document, we can multiply all this by a related term and its reciprocal. The complete probability calculation is thus factored out into the following four terms (Term (2) simplifies to the final form shown to the right of the arrow):

$$\prod_{j:q_j=1,a_j(d)>0} \frac{P\left(A_j = a_j\left(d\right)|R=y\right)}{P\left(A_j = a_j\left(d\right)\right)} \tag{1}$$

$$\prod_{j:q_j=1,a_j(d)=0} \frac{P\left(A_j = a_j\left(d\right)|R=y\right)}{P\left(A_j = a_j\left(d\right)\right)} \rightarrow \prod_{j:q_j=1,a_j(d)=0} \frac{P\left(A_j = 0|R=y\right)}{P\left(A_j = 0\right)} \tag{2}$$

$$\prod_{j:q_j=1,a_j(d)>0} \frac{P\left(A_j = 0|R=y\right)}{P\left(A_j = 0\right)} \tag{3}$$

$$\prod_{j:q_j=1,a_j(d)>0} \frac{P\left(A_j = 0\right)}{P\left(A_j = 0|R=y\right)} \tag{4}$$

The product of Terms (2) and (3) is not dependent on d and hence can be left out of the ranking function. Therefore after combining (1) and (4), we end up with the following ranking function.

$$\prod_{j:q_j=1,a_j(d)>0} \frac{P\left(A_j = a_j\left(d\right)|R=y\right)}{P\left(A_j = a_j\left(d\right)\right)} \cdot \frac{P\left(A_j = 0\right)}{P\left(A_j = 0|R=y\right)}$$

We can think of this as a "sorta" version of the Robertson/Spärck Jones Model ('76).

Now we have a model/paradigm to work with. We just need to estimate the values for all the terms in the model to get a ranking function.

3 Estimation Models

3.1 Binary estimate - Croft and Harper

This model was put forward by Croft and Harper in 1979 as a means of estimation for cases where a lack of relevance information means that the quantities put forth in the Robertson/Spärck Jones Model can't be evaluated.

Assumption: We will assume that the A_j attributes are binary (i.e., they represent a term being either absent or present; no weighting is involved). Therefore our probability estimation $\hat{P}$ is

simplified considerably, and

$$\begin{aligned}
\hat{P}(A_j = 1) &= \frac{no.\ of\ docs\ containing\ v^{(j)}}{no.\ of\ docs\ in\ C} = \frac{N^{(j)}}{N} \\
\hat{P}(A_j = 0) &= 1 - \hat{P}(A_j = 1) \\
\hat{P}(A_j = 1 | R = y) &= constant
\end{aligned}$$

(However, this implies that $\hat{P}(A_j = 1|R = y)$ is the same for all terms, which is probably not the case.) Therefore

$$\begin{aligned}
Score(d) &= \prod_{j:q_j=1,a_j(d)=1} \frac{constant}{1 - constant} \cdot \frac{1 - \frac{N^{(j)}}{N}}{\frac{N^{(j)}}{N}} \\
Score(d) &= \prod_{j:q_j=1,a_j(d)=1} \frac{constant}{1 - constant} \cdot \frac{N - N^{(j)}}{N^{(j)}}
\end{aligned}$$

It is interesting to note that this expression "sort of" looks like the **IDF** expression we came up with in a previous lecture. We take the log of the expressesion to turn our product into a summation, and insert q_j and a_j (which is mathematically justified for binary values). Hence,

$$\sum_j q_j \cdot a_j(d) \cdot \log\left(\frac{N - N^{(j)}}{N^{(j)}}\right)$$

where N is the number of documents in our corpus, and $N^{(j)}$ is the number of documents containing term $v^{(j)}$.

There are a few interesting points to note about this expression:

- It looks very much like the VSM scheme.
- Some people consider it to be a derivation or a justification of the IDF.
- The term frequency and the length normalization expressions are missing from the expression. This is not surprising considering our original assumption of binary A_j's.
- This method was used for a while after it was developed. However, it was useful primarily because the documents it was applied to were comprised of term lists, where the appearance of a term could actually *be* a case of "there" or "not there."

3.2 Poisson-Based Approach

[Bookstein/Swanson '74, Harter '75, Robertson/Walker '94] The Poisson function is a model of arrival times. *We will use it to model the number of times a particular term occurs in a document of a given length*, where before we only considered whether a term was present or absent. The Poisson distribution is given as follows:

$$Pois_\mu(X = x) = \frac{\mu^x}{x!} \cdot e^{-\mu}$$

i.e for a given document length, what is the probability that we will see the a given term x times (with an *expected* term count μ).

As a prelude to next time, we'll consider the following approach, set up as follows:

$$\begin{aligned} Expected\,Value\,E\left(X\right) &= \mu \\ Pois_{\mu}\left(X=0\right) &= e^{-\mu} \\ P\left(A_j = a_j\left(d\right)|R=y\right) &\rightarrow Poisson\,with\,parameter\,\rho \\ P\left(A_j = a_j\left(d\right)\right) &\rightarrow Poisson\,with\,parameter\,\gamma \end{aligned}$$

Assumption: $\rho > \gamma$ because $q_j = 1$ (i.e., the attribute term is in the query, so the probability that the term appears in a document selected from the "relevant" pool is higher the probability of it appearing from the pool of all documents). Note also that, really, $\rho = \rho_j$ and $\gamma = \gamma_j$ – they're term-dependent. We're just omitting the subscripts for convenience.

Hence based on this model, the RSJ quantity turns out to be:

$$Score(d) = \frac{\left\{\frac{\rho^{a_j(d)}}{[a_j(d)]!}\cdot e^{-\rho}\right\}\left\{e^{-\gamma}\right\}}{\left\{\frac{\gamma^{a_j(d)}}{[a_j(d)]!}\cdot e^{-\gamma}\right\}\left\{e^{-\rho}\right\}} = \left(\frac{\rho_j}{\gamma_j}\right)^{a_j(d)}$$

Taking the log of the RSJ model will convert it into a summation:

$$Score(d) = \sum_{j:a_j(d)>0, q_j=1} a_j\left(d\right)\cdot\log\left(\frac{\rho_j}{\gamma_j}\right)$$

In the above, $a_j(d)$ represents our term frequency; what can we do with the remaining term? Can it be considered to be a form of IDF?

4 Questions: Investigating the Binary and 1-Poisson Models

For these exercises, let's assume that we have a fixed query q = "information retrieval." We also have a corpus consisting of four documents, each 12 words long:

- D_1 = "Information retrieval aspires to get the right information for the right person."
- D_2 = "My dog's retrieval was greatly helped by the information from the neighbors."
- D_3 = "Despite having all this information, I still can't understand a single thing!"
- D_4 = "What terms does this sentence have in common with the other three?"

Notice that the order the documents appear in happens to correspond to their desired ranking: D_1's content is closest to the sense of the query, while D_4 has no direct connection to the query at all.

4.1 Applying the derived functions

To start, let's calculate the rankings using our Croft-Harper variant of the RSJ model,

$$Score(d) = \sum_j q_j \cdot a_j(d) \cdot \log\left(\frac{N - N^{(j)}}{N^{(j)}}\right)$$

Since our term "counts" are binary – present or absent – each quantity of the summation reduces to zero except for terms that are present in both the query and the document. Using the corpus and query above,

$$N = 4$$
$$N^{(1)} = 3$$
$$N^{(2)} = 2$$

where $v^{(1)}$ = "information" and $v^{(2)}$ = "retrieval".

Now we can calculate our score for each document:

$$S_1 = \log\left(\frac{4-3}{3}\right) + \log\left(\frac{4-2}{2}\right) = \log\left(\frac{1}{3}\right) + log(1) = -0.477$$
$$S_2 = \log\left(\frac{1}{3}\right) + \log(1) = -0.477$$
$$S_3 = \log\left(\frac{1}{3}\right) + 0 = -0.477$$
$$S_4 = 0 + 0 = 0$$

In this case the binary solution discarded the one completely irrelevant document, but ranked the other three equally![1] Can we do better than this?

Our Poisson scoring function was derived as

$$Score = \sum_{j:a_j(d)>0, q_j=1} a_j(d) \cdot \log\left(\frac{\rho_j}{\gamma_j}\right)$$

Here, ρ is the expected value (for term frequency) among relevant documents, and γ is the expected value among all documents. For the purposes of this example, we will make the somewhat unrealistic assumption that we can calculate reasonable expected values solely from our four-document corpus. If we consider D_1 to be the only truly relevant document, then, by estimating $\rho(j)$ and $\gamma(j)$ as the number of occurrences of $v^{(j)}$ in the set of relevant (i.e., all) documents divided by the total number of documents in the relevant set, we get

$$\rho^{(1)} = 2 \div 1 = 2$$
$$\rho^{(2)} = 1 \div 1 = 1$$
$$\gamma^{(1)} = 4 \div 4 = 1$$
$$\gamma^{(2)} = 2 \div 4 = \frac{1}{2}$$

[1] This occurs because the function does not properly account for terms that appear in a majority of our documents. A term that appears in *all* documents has a weight of $(\log 0)$, or $-\infty$!

Once we have values for ρ_j and γ_j, calculation of our ranks is straightforward:

$$S_1 = 2 \cdot \log\left(\frac{2}{1}\right) + 1 \cdot \log\left(\frac{1}{\frac{1}{2}}\right) = 0.90$$
$$S_2 = 1 \cdot \log\left(\frac{2}{1}\right) + 1 \cdot \log\left(\frac{1}{\frac{1}{2}}\right) = 0.60$$
$$S_3 = 1 \cdot \log\left(\frac{2}{1}\right) + 0 = 0.30$$
$$S_4 = 0 + 0 = 0$$

Notice that the Poisson estimation ranked our documents correctly. Also, in this case, D_1 and D_2 do *not* produce the same score. However, the weights of the IDF-ish function was the same in both cases, so the difference is entirely due to term frequency.

4.2 Extending the derived functions

Recall that using the Poisson approximation was justified by asserting that the documents being considered were of a fixed length. Is it possible to modify our method to apply this ranking function to documents of mixed length?

For this experiment, we'll consider a fifth document:

- D_5 = "Information is power."

This text is much shorter than the other items in the corpus. We'll create a new corpus consisting of just two documents, D_1 and D_5; our query remains the same. There are two approaches we can take to "equalizing" the two document lengths: we can either increase the length of D_5 or reduce the length of D_1.

To use the length of D_1, we can pad D_5 with "junk" words that don't appear in our query. For example:

- D_6 = "Information is power junk junk junk junk junk junk junk junk junk"

Provided that these terms do not appear in the query, they will reduce to zero in the summation.

If we then apply our ranking function, we find that

$$\rho^{(1)} = 2$$
$$\rho^{(2)} = 1$$
$$\gamma^{(1)} = 1$$
$$\gamma^{(2)} = \frac{1}{2}$$

and we get

$$S_1 = 2 \cdot \log(2) + \log(2) = 0.90$$
$$S_2 = \log(2) = 0.30$$

which is a satisfactory result.

Reducing the length of D_1 is not so straightforward. We will begin by splitting the original document into "sub-documents" of the same length as D_5:

- $D_{1,0}$ = "Information retrieval aspires"
- $D_{1,1}$ = "to get the"
- $D_{1,2}$ = "right information for"
- $D_{1,3}$ = "the right person"

There are now five documents in our collection. Before we can apply the ranking function, however, we must first consider which of these documents, or how many of them, are now "relevant." Although it is mathematically suspicious, we will first consider the case where for purposes of relevancy we claim there are still only two documents in the corpus, but calculate the scoring function for five. Thus,

$$\rho^{(1)} = \frac{2}{1} = 2 \text{ (Two occurrences in the one relevant document)}$$

$$\rho^{(2)} = \frac{1}{1} = 1 \text{ (One occurrence in the one relevant document)}$$

$$\gamma^{(1)} = \frac{3}{2}$$

$$\gamma^{(2)} = \frac{1}{2}$$

and

$$S_{1,0} = \log\left(\frac{4}{3}\right) + \log(2) = 0.43$$

$$S_{1,1} = 0$$

$$S_{1,2} = \log\left(\frac{4}{3}\right) + 0 = 0.12$$

$$S_{1,3} = 0$$

$$S_5 = \log\left(\frac{4}{3}\right) + 0 = 0.12$$

In this case, once the ranking function has returned its best pick of sub-documents – $D_{1,0}$ – we must re-associate it with its original document, D_1. Once again we have arrived at the most relevant document. (Again, though, notice that two of the sub-documents received exactly the same score.)

As a final case, we'll reconsider our organization of relevant documents. We proceed with the same five sub-documents, but this time we assert that there are *four* relevant documents in our collection of five, and make our calculations from that basis:

$$\rho^{(1)} = \frac{2}{4} = \frac{1}{2}$$

$$\rho^{(2)} = \frac{1}{4}$$

$$\gamma^{(1)} = \frac{3}{5}$$
$$\gamma^{(2)} = \frac{1}{5}$$

$$S_{1,0} = \log\left(\frac{5}{6}\right) + \log\left(\frac{5}{4}\right) = 0.01$$
$$S_{1,1} = 0$$
$$S_{1,2} = \log\left(\frac{5}{6}\right) = -0.08$$
$$S_{1,3} = 0$$
$$S_5 = \log\left(\frac{5}{6}\right) = -0.08$$

Although our quantitative ranking values have changed, the relative distribution of values remains unchanged, and the same sub-document – and thus main document – is returned as the most relevant. Once more, two sub-documents are scored identically.

As a final point of interest, notice that the original Poisson ranking of equal-length documents in Section 4.1 gave equal weight to the two search terms, "information" and "retrieval." However, in our subdivided ranking calculation, the second term receives a higher weight than term one: 1.2 versus 0.83. This is reminiscent of an IDF-term behavior; does it hold for more generalized cases?

4.3 Further thinking

Ideas for further consideration:

1. The binary scoring function had been used for a time, and was effective, because it was applied to term lists where presence or absence was, in fact, a yes or no proposition; it was suggested in lecture that this effect might apply as well to very short documents. Would it be possible to use the subdivision technique applied in Section 4.2 to rank longer documents this way?

2. Can we make the relative order of terms in a query part of the relevance measure or document ranking scheme? Most of the techniques we have seen up to this point generally deal with the *presence* of a term in a document, but not necessarily its *position*. Maybe we can achieve better ranking results if we take into account the relative positions of search terms within a document.

   One way of doing this could be to extend the arguments of an attribute vector to include the average location of a term in addition to its frequency. Therefore, if $al^{(j)}$ is the average position of the jth term in a document, and $tf^{(j)}$ is the term frequency of the jth term, then

   $$A \rightarrow (\{tf^{(1)}, al^{(1)}\}, \{tf^{(2)}, al^{(2)}\}, ..., \{tf^{(j)}, al^{(j)}\})$$

   Thus, if we are searching for the terms "information retrieval", within a relevant document the average location of "information" will be less than that of "retrieval." Of course, this may not necessarily hold true in all cases, but it may give us further information beyond a term's presence or absence.