CS630 Lecture Notes

Lecturer: Lillian Lee
Scribes: Chris Danis (cgd3) & Brian Rogan (bcr6)

Lecture 7: 16 February 2006

1 Review & introduction

Today we will be covering the language modeling (or LM) approach to IR. We'll start with the motivation and continue on to a specific example. LM is also a probabilistic approach; however, it differs from "classic" probabilistic model. The RSJ model had nice motivations, but we couldn't actually provide satisfactory estimates for it in practice; let's go back to first principles. We'll be following the approach of Lafferty and Zhai '03 ([1]).

In the LM approach, queries are explicitly modeled as an unknown in the equations – unlike the RSJ approach, where the query was implicitly assumed to be fixed to a known value. Thus we need to shift our previous notation to account for this distinction. Written below is our RSJ ranking function expressed in our new notation. Our probabilistic retrieval score $P_q(R = y|D = d)$ after a Bayes flip is:

$$\frac{P_q(D = d|R = y)P_q(R = y)}{P_q(D = d)}.$$

$P_q(R = y)$ is constant across documents and can be removed under ranking. Also, we can ignore the q in $P_q(D = d)$ because as the documents in our corpus have already been written at the time the user enters the query, it seems reasonable to assume they are independent. Thus the final scoring function is

$$\frac{P_q(D = d|R = y)}{P(D = d)}.$$

2 The language modeling approach

One of the motivations behind the LM approach is to increase the importance of the query. Consider a new scoring function:

$$P(R = y|D = d, Q = q).$$

Why should we bother? On first blush, this looks like a minor semantic change. (We'll call this the LM ranking function even though we haven't explained why yet.)

2.1 Do you get better rankings out of this?

2.1.1 Answer 1

No, we don't. We get the same rankings; we just have to slide around the term for Q.

We start with $P(R = y|D = d, Q = q)$. Let's Bayes flip R & D, resulting in

$$\frac{P(D = d|R = y, Q = q)P(R = y|Q = q)}{P(D = d|Q = q)}.$$

$P(R = y|Q = q)$ falls out under ranking as it is independent of the document. Eliminate it.

There's no notion of relevance in $P(D = d|Q = q)$. So how can they possibly depend on each other? Assume D is independent of Q. So we're left with $P(D = d)$ on the bottom. This is exactly our original ranking function: $\frac{P_q(D=d|R=y)}{P(D=d)}$.

This provides a post hoc justification of the work of Ponte & Croft '98 ([2]).

2.1.2 Answer 2

But the above isn't the only thing we can do. What happens if we instead Bayes-flip R and Q in $P(R = y|D = d, Q = q)$? We get:

$$\frac{P(Q = q|R = y, D = d)P(R = y|D = d)}{P(Q = q|D = d)}.$$

Q is now on the left-hand side of the conditional bar. We already assumed D & Q to be independent before; let's do that again. If Q is independent of D, then $P(Q = q|D = d) = P(Q = q)$, and this term falls out under rank; it has *nothing* to do with the document. We're left with

$$P(Q = q|R = y, D = d)P(R = y|D = d).$$

So what do we do with $P(R = y|D = d)$? This term seems strange at first. However, it's simply the probability of relevance *a priori* without seeing a query. Consider: for the majority of queries, is a document from the Encyclopedia Britannica likely to be more relevant than an entry in a random Internet denizen's blog? This *a priori* relevance term can be applied in many different ways. If you simply don't want to worry about assigning *a priori* relevance scores, you can assume it's a constant, or at least that R and D are independent (then $P(R = y|D = d) = P(R = y)$ which doesn't affect ranking)[1].

We're left to explain the term $P(Q = q|R = y, D = d)$. This also looks strange – it doesn't "feel" like what we've been considering in past models. If we're scoring by this alone, we "prefer document d if, given d and the fact that d is relevant, query q was issued". Essentially: if you thought the document

[1] This assumption is reasonable in this context; however, this is not so with our original scoring function $P_q(R = y|D = d)$: it seems rather bizarre to say R and D are independent given that you know the query.

was relevant, consider the document as evidence of what information the user wants.

Here we have an inference scenario. Let's suppose we have a class of models of all information needs for our users, and some function $t(d)$ defining a mapping from a document d to our space of information needs or (more loosely) "topics".

1. Assume the document d is relevant and infer the model $t(d)$ as the information need the document provides.

2. Check whether the user is likely to issue query q if $t(d)$ were their information need.

This is quite a departure from our previous approaches. Looking at a broader view of both probabilistic approaches we've considered, we now have $P(Q = q|R = y, D = d) \rightarrow P_{t(d)}(q)$: the probability of generating the user's query given a model induced from the document in question. Contrast this with our original $\frac{P(D=d|R=y,Q=q)}{P(D=d)} \rightarrow \frac{P_{t(q)}(d)}{P(D=d)}$: the probability of generating a document under a model based on the user's query.

Lafferty and Zhai call this distinction the "importance of being reversed"[2]. Why should we prefer this new model?

- It allows for offline computation of information need models (all we need is a corpus; $t(d)$ is computed sans query);

- There's more data (and possibly richer data) in the documents than in queries;

- There's no hidden random variable for relevance, which, in situations in which no relevance information is available, simplifies the estimation problem considerably.

3 Specific example

A very simple model is used in this example: multinomial "topic" models with corpus-dependent Dirichlet priors[3].

Our generation scenario: to generate a string[4] of length l, do l independent rolls of an m-sided die (with a term on each side). We do *not* assume terms are equiprobable. $\vec{\theta} = (\theta_1, \cdots, \theta_m)$; let θ_j = the probability of the term $v^{(j)}$ appearing. Of course, $\sum_j \theta_j(d) = 1$ must hold.

Given a document d, we define $P_{t(d)} \equiv \vec{\theta}(d)$. We need values for $\theta_j(d)$; let's assume for now that they're known (we'll form an estimate later). If so, what is $P_{\vec{\theta}(d)}(q)$?[5] Let's also rewrite the query q as a vector: $\vec{q} = (tf_1(q), \cdots, tf_j(q), \cdots, tf_m(q))$

[2] [1], page 7.

[3] This is from Zhai & Lafferty at SIGIR '01; see [3].

[4] We consider a string to be a sequence of terms, which could be a query, a document, or an entire corpus.

[5] When the probability is with respect to all strings of the same length as q.

Given this notation and set-up, let's score as follows:

$$P_{t(d)}(q) \equiv P_{\vec{\theta}(d)}(\vec{q}) = f(\vec{q}, d) \cdot k(\vec{q}) \,.$$

$f(\vec{q}, d)$ is the probability of observing the "sorted version" of q (i.e. q with the terms in the order they appear in V). $f(\vec{q}, d) = \prod_j [\theta_j(d)]^{tf_j(q)}$.

$k(\vec{q})$ is simply the number of permutations of the sorted version of q.

This leaves us with the question: what is $\theta_j(d)$? We can estimate it as

$$\frac{tf_j(d) + \mu \times \frac{tf_j(C)}{|C|}}{|d| + \mu}.$$

$|d| = \sum_j tf_j(d)$; the μ term is needed for normalization purposes (without it, putting just $\frac{tf_j(C)}{|C|}$ in the numerator would result in $\sum_j \theta_j \neq 1$).

Here we have a tf_j term and a length normalization term: two of the three elements we'd like to see in an IR system. Unfortunately, the $\mu \times \frac{tf_j(C)}{|C|}$ term looks rather anti-idf. As a generative model this makes sense: we'd like to generate terms we see more often. As a ranking model, however, this looks like a problem: we're going to prefer to match terms like "the". This issue will be resolved next lecture.

References

[1] John Lafferty and ChengXiang Zhai. *Language Modeling for Information Retrieval*, chapter Probabilistic Relevance Models Based on Document and Query Generation, pages 1–10. Kluwer Academic Publishers, 2003.

[2] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Research and Development in Information Retrieval*, pages 275–281, 1998.

[3] ChengXiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Research and Development in Information Retrieval*, pages 334–342, 2001.