Notes on the KL-divergence retrieval formula and Dirichlet prior smoothing

ChengXiang Zhai

March 11, 2007

1 The KL-divergence measure

Given two probability mass functions \( p(x) \) and \( q(x) \), \( D(p \| q) \), the Kullback-Leibler divergence (or relative entropy) between \( p \) and \( q \) is defined as

\[
D(p \| q) = \sum_x p(x) \log \frac{p(x)}{q(x)}
\]

It is easy to show that \( D(p \| q) \) is always non-negative and is zero if and only if \( p = q \). Even though it is not a true distance between distributions (because it is not symmetric and does not satisfy the triangle inequality), it is still often useful to think of the KL-divergence as a “distance” between distributions [Cover and Thomas, 1991].

2 Using KL-divergence for retrieval

Suppose that a query \( q \) is generated by a generative model \( p(q \mid \theta_Q) \) with \( \theta_Q \) denoting the parameters of the query unigram language model. Similarly, assume that a document \( d \) is generated by a generative model \( p(d \mid \theta_D) \) with \( \theta_D \) denoting the parameters of the document unigram language model. If \( \hat{\theta}_Q \) and \( \hat{\theta}_D \) are the estimated query and document language models respectively, then, the relevance value of \( d \) with respect to \( q \) can be measured by the following negative KL-divergence function [Zhai and Lafferty, 2001a]:

\[
-D(\hat{\theta}_Q \| \hat{\theta}_D) = \sum_w p(w \mid \hat{\theta}_Q) \log p(w \mid \hat{\theta}_D) + \left( -\sum_w p(w \mid \hat{\theta}_Q) \log p(w \mid \hat{\theta}_Q) \right)
\]

Note that the second term on the right-hand side of the formula is a query-dependent constant, or more specifically, the entropy of the query model \( \hat{\theta}_Q \). It can be ignored for the purpose of ranking documents. In general, the computation of the above formula involves a sum over all the words that have a non-zero probability according to \( p(w \mid \hat{\theta}_Q) \). However, when \( \hat{\theta}_D \) is based on certain general smoothing method, the computation would only involve a sum over those that both have a non-zero probability according to \( p(w \mid \hat{\theta}_Q) \) and occur in document \( d \). Such a sum can be computed much more efficiently with an inverted index.

We now explain this in detail. The general smoothing scheme we assume is the following

\[
p(w \mid \hat{\theta}_D) = \begin{cases} 
p_s(w \mid d) & \text{if word } w \text{ is seen} \\
\alpha_d p(w \mid C) & \text{otherwise} \end{cases}
\]
where \( p_s(w \mid d) \) is the smoothed probability of a word seen in the document, \( p(w \mid C) \) is the collection language model, and \( \alpha_d \) is a coefficient controlling the probability mass assigned to unseen words, so that all probabilities sum to one. In general, \( \alpha_d \) may depend on \( d \). Indeed, if \( p_s(w \mid d) \) is given, we must have

\[
\alpha = \frac{1 - \sum_{w : c(w, d) > 0} p_s(w \mid d)}{1 - \sum_{w : c(w, d) > 0} p(w \mid C)}
\]

Thus, individual smoothing methods essentially differ in their choice of \( p_s(w \mid d) \).

The collection language model \( p(w \mid C) \) is typically estimated by \( \frac{c(w, C) + 1}{V + \sum_{w'} c(w', C)} \), where \( V \) is an estimated vocabulary size (e.g., the total number of distinct words in the collection). One advantage of the smoothed version is that it would never give a zero probability to any term, but in terms of retrieval performance, there will not be any significant difference in these two versions, since \( \sum_{w'} c(w', C) \) is often significantly larger than \( V \).

If can be shown that with such a smoothing scheme, the KL-divergence scoring formula is essentially:

\[
\sum_{w : c(w, d) > 0, p(w \mid \tilde{Q}) > 0} p(w \mid \tilde{Q}) \log \frac{p_s(w \mid d)}{\alpha_d p(w \mid C)} + \log \frac{\mu}{\mu + |d|}
\]

Note that the scoring is now based on a sum over all the terms that both have a non-zero probability according to \( p(w \mid \tilde{Q}) \) and occur in the document, i.e., all “matched” terms.

### 3 Using Dirichlet prior smoothing

Dirichlet prior smoothing is one particular smoothing method that follows the general smoothing scheme mentioned in the previous section. In particular,

\[
p_s(w \mid d) = \frac{c(w, d) + \mu p(w \mid C)}{|d| + \mu}
\]

and

\[
\alpha_d = \frac{\mu}{\mu + |d|}
\]

Plugging these into equation 1, we see that with Dirichlet prior smoothing, our KL-divergence scoring formula is

\[
\sum_{w : c(w, d) > 0, p(w \mid \tilde{Q}) > 0} p(w \mid \tilde{Q}) \log (1 + \frac{c(w, d)}{\mu p(w \mid C)}) + \log \frac{\mu}{\mu + |d|}
\]

This is the retrieval formula that you are asked to implement in assignment 3. \( p(w \mid \tilde{Q}) \) is passed into the function `computeWeight` as an argument. This is where the code is different from that in assignment 2 where the same argument carries the query term frequency. In the simplest case (i.e., initial retrieval), the probability passed in is just the normalized query term frequency (i.e., \( c(w, q) / |q| \)).
4 Computing the query model $p(w|\hat{\theta}_Q)$

You may be wondering how we can compute $p(w|\hat{\theta}_Q)$. This is exactly where the KL-divergence retrieval method is better than the simple query likelihood method – we can have different ways of computing it!

The simplest way is to estimate this probability by the maximum likelihood estimator using the query text as evidence, which gives us

$$p_{ml}(w|\hat{\theta}_Q) = \frac{c(w, q)}{|q|}$$

Using this estimated value, you should see easily that the KL-divergence scoring formula is essentially the same as the query likelihood retrieval formula as presented in [Zhai and Lafferty, 2001b].

Question 1 in assignment 3 asks you to evaluate such a simple query model, which is equivalent to the query likelihood method.

A more interesting way of computing $p(w|\hat{\theta}_Q)$ is to exploit feedback documents. Specifically, we can interpolate the simple $p_{ml}(w|\hat{\theta}_Q)$ with a feedback model $p(w|\theta_F)$ estimated based on feedback documents. That is,

$$p(w|\hat{\theta}_Q) = (1 - \alpha)p_{ml}(w|\hat{\theta}_Q) + \alpha p(w|\theta_F)$$  \hspace{1cm} (3)

where, $\alpha$ is a parameter that needs to be set empirically. Please note that this $\alpha$ is different from $\alpha_d$ in the smoothing formula.

Of course, the next question is how to estimate $p(w|\theta_F)$? One approach is to assume the following two component mixture model for the feedback documents, where one component model is $p(w|\theta_F)$ and the other is $p(w|C)$, the collection language model.

$$\log p(F|\theta_F) = \sum_{i=1}^{k} \sum_w c(w; d_i) \log((1 - \lambda)p(w|\theta_F) + \lambda p(w|C))$$

where, $F = \{d_1, ..., d_k\}$ is the set of feedback documents, and $\lambda$ is yet another parameter that indicates the amount of “background noise” in the feedback documents, and that needs to be set empirically. Now, given $\lambda$, the feedback documents $F$, and the collection language model $p(w|C)$, we can use the EM algorithm to compute the maximum likelihood estimate of $\theta_F$. That is, the estimated $\theta_F$ is

$$\hat{\theta}_F = \arg \max_{\theta_F} \log p(F|\theta_F)$$

The EM updating formulas are:

$$z^{(n)}(w) = \frac{(1 - \lambda)p_{\lambda}^{(n)}(w|\theta_F)}{(1 - \lambda)p_{\lambda}^{(n)}(w|\theta_F) + \lambda p(w|C)}$$

$$p_{\lambda}^{(n+1)}(w|\theta_F) = \frac{\sum_{j=1}^{k} c(w; d_j)z^{(n)}(w)z^{(n)}(w_i)}{\sum_{i=1}^{k} \sum_{j=1}^{k} c(w_i; d_j)z^{(n)}(w_i)}$$

Question 3 in assignment 3 asks you to complete the implementation of such an EM algorithm. All the questions after that refer to feedback, which means computing $p(w|\hat{\theta}_Q)$ with formula 3.
References

