INFO 630 / CS 674 Lecture Notes

The Language Modeling Approach to Information Retrieval

Lecturer: Lillian Lee
Lecture 9: September 25, 2007
Scribes: Vladimir Barash, Stephen Purpura, Shaomei Wu

Introduction and Motivation

Today's lecture notes cover an introduction to the application of statistical language modeling to information retrieval as motivated by "The Language Modeling Approach to Information Retrieval" by Ponte and Croft from SIGIR '98. Language modeling is the 3rd major paradigm that we will cover in information retrieval. At the time of application, statistical language modeling had been used successfully by the speech recognition community and Ponte and Croft recognized the value of adapting the method to information retrieval. You can consider our derivation as a simplified post-hoc justification for their approach (from [Lafferty and Zhai, '03]). Before we dive into the details, we will review the beginning of our derivation from the previous class.

(Re-) Starting Point: Probabilistic Modeling

Given Q as a random variable over queries (based on a user's information need), D as a random variable over documents (based on authors/corpus) and R as a binary relevance variable, the Probabilistic Modeling approach scores the relevance between document and query by Equation 0.

$$(Eq\,0):\ \ Score(q,d)=P(R=y|D=d,Q=q)$$

A Bayes flip of Q and R in Equation 0 turns it into *P(Q=q|R=y,D=d)P(R=y|D=d)/P(Q=q|D=d)*. Knowing that the query Q is given independently of document D, the term *P(Q=q|D=d)* becomes *P(Q=q)*, which is document-independent and so can be ignored when we are looking at ranking documents by relevance. In this way, the relevance scoring function given in Equation 0 can be turned into Equation 1.

$$(Eq\,1):\ \ P(Q=q|R=y,D=d)\,P(R=y|D=d)$$

To conclude the introduction above, in the probabilistic modeling paradigm, we started with a "Bayes Flip" to swap *"R = y"* and *"D = d"* in Equation 0, which transformed it into the probability of a certain document d having (or not having) specific attributes A_j, given that d is relevant to query q. The Language Modeling approach performs a different Bayes Flip on Equation 0, transforming it into the probability of a query q, given some document d that is relevant to q. By explicitly shifting our notation, we have intentionally given considerably more weight to the importance of the query.

Justification: One Derivation of $P_{t(d)}(q)$

> ***"... in our view, users have a reasonable idea of terms that are likely to occur in documents of interest and will choose query terms that distinguish these documents from others in the collection ..." -- [Ponte & Croft, SIGIR '98]***

In the probabilistic approach of RSJ, the query was implicitly assumed to be fixed. By transforming the classic probabilistic model into Equation 1 given above, [Ponte and Croft, SIGIR '98] (through [Lafferty and Zhai, '03]'s post-hoc justification) models the relevant-document retrieval process by

acknowledging that a user issues each query. But the key to this approach is the notion of a "language model", which can be considered a probabilistic source for text strings, a.k.a. a "text generator" or "probability assigner". So equation 1 can (ignoring the second quantity) be interpreted as scoring highly those documents such that the "query probability" is high. And a way of further refining this interpretation is: if a document has as source the same language model as that which generated the query, then that document should be ranked highly.

More specifically, suppose that there exists a finite set of "topic" language models, where each topic represents some information need (for the author), each of which generates documents under that topic. Then one interpretation of the job of a Language Modeling information retrieval system is, given a query *q*, for any document *d* in the corpus, assuming it is relevant, to try to infer the topic language model for it (LM *t(d)* given *T = topic model*). Because the document can be considered to be more relevant to the given query if they are (generated) under the same topic, after inferring the language model for each document, the system checks its assumption of relevancy against the query by calculating $P_{t(d)}(q)$; this quantity stands for the probability of *q* being generated from the same language model as *d*.

From our new proposed language model perspective, we now attempt to use the notion that relevance (*"R = y"*) is equivalent to the notion that the query, *q,* and the document, *d,* have an identical topic language model (LM). To facilitate this demonstration, we formally introduce T_D and T_Q - random variables standing for the topic LM for the document and the query, respectively. Then, our scoring function becomes:

$$(Eq\,2):\ \sum_{t,t'} P(Q=q, T_D=t, T_Q=t' | R=y, D=d) P(R=y|D=d)$$

Given that *R=y*, terms for all cases where *t* and *t'* are not the same disappear.

$$(Eq\,3):\ \sum_{t} P(Q=q, T_D=t, T_Q=t | R=y, D=d) P(R=y|D=d)$$

We would like to drop the conditioning on R=y in equation 3, as it is redundant after including $T_D = T_Q$ in the formula. If we dropped the conditioning right now, however, the first term of the probability product would not sum to 1 over all t. In order to get rid of R=y, we rephrase the probability as a joint probability of Q=q, T_D=t, T_Q=t, and R=y; after rephrasing, the equation is balanced out by dividing it by the probability *R=y|D=d*:

$$(Eq\,4):\ \sum_{t} \frac{P(Q=q, T_D=t, T_Q=t, R=y|D=d) P(R=y|D=d)}{P(R=y|D=d)}$$

Now we simplify and (since we're now taking a joint probability!) we drop "R=y" from the first term as redundant:

$$(Eq\,5):\ \sum_{t} P(Q=q, T_D=t, T_Q=t|D=d)$$

Have we lost *"R = y"*? No. Relevance is implicitly encoded in the joint probability!

We want to further transform equation 5 to condition on our source models T_D and T_Q. T_Q tells us how to generate queries, so we can calculate what $P(Q=q\ |T_Q=t)$ for any *q*. To do this, use Bayes' rule:

$$(Eq\,6):\ \sum_{t} P(Q=q|T_D=t, T_Q=t, D=d) P(T_D=t, T_Q=t|D=d)$$

Because the way a user gives queries is not affected by documents or corpus, given the query model, in Equation 6, $Q=q$ is independent of $T_D=t$ (the topic LM generating the document) given the query model and $D=d$ (the document itself), given the query model $T_Q=q$:

$$(Eq7):\ \sum_t P(Q=q|T_Q=t)P(T_D=t,T_Q=t|D=d)$$

The first probability term in Eq 7 looks good, but the second one is still too complicated. Let's simplify it by transforming the term into a probability of $T_Q=t$ conditioned on $T_D=t$, $D=d$.

$$(Eq8):\ P(T_D=t,T_Q=t|D=d) = P(T_Q=t|T_D=t,D=d)P(T_D=t|D=d)$$

We assume that T_Q is chosen independently of T_D and d, so $T_Q=q$ is independent of these two quantities. Then, equation 7 becomes:

$$(Eq9):\ \sum_t P(Q=q|T_Q=t)P(T_Q=t)P(T_D=t|D=d)$$

We cannot simplify equation 9 without further assumptions. In this case, we assume that estimation will be easier if we don't have to deal with all possible topics. So, using an assumption conceptually similar to a Viterbi approximation, we assume that there exists some *t*(d)* such that:

$$(Eq10):\ P(T_D=t^*(d)|D=d)=1$$

Then equation 9 becomes:

$$(Eq11):\ P(Q=q|T_Q=t^*(d))P(T_Q=t^*(d))$$

as the term $P(T_D=t \mid D=d)$ is 0 for all *t(d)* other than *t*(d)*, and *1 for t*(d*).

Secondly, we assume that the selection of T_Q is uniform (an ok assumption given a large number of users with many different information needs). Then $P(T_Q=t^*(d))$ is a document-independent constant and equation 11 becomes:

$$(Eq12):\ P(Q=q|T_Q=t^*(d))$$

Now equation 12 is equivalent to $P_{t(d)}(q)$! The query is generated by the same source as the documents.

For alternative derivations of the LM model, see [Lafferty & Zhai, SIGIR '01] (their model features a risk/loss-based framework), and [Lavrenko & Croft, SIGIR '01] (who do not remove the relevance term from their equations, but instead estimate it using a pseudo-feedback method).

Multinomial Language Modeling with Dirichlet Smoothing

A large number of very sophisticated Language Model technologies have been developed, any of which can be used to calculate $P_{t(d)}(q)$. However, as queries are often simple and agrammatical collections of terms, it makes sense to use a simple probabilistic model in this case. One such model is a "multinomial LM with Dirichlet smoothing" [Zhai & Lafferty SIGIR '01]

Given a query, *q*, we assume the random choice for *Q* is over all *q'* of the same length *L* as *q*.

Let's define some useful vector notation first: (vector quantities are in **bold** outside of the equations)

$$(Eq\,13):\ \vec{x}=\begin{bmatrix} x[1] \\ \dots \\ x[m] \end{bmatrix}=(x[1],\dots,x[m])^T$$

$$(Eq\,14):\ \vec{x}[\,.]=\sum_{j=1}^{m} x[j]$$

Here, *m* is the size of entire vocabulary. We will be analyzing term count vectors for the query, documents and corpus, which are denoted as $\vec{q}$, $\vec{d}$, and $\vec{c}$, respectively. Note that *d[.]* is simply the document length in words, including repeats.

Call the multinomial parameters induced from a particular document *d* the vector θ_d.

$\theta_d[j]$ = the probability of choosing v_j from V for a word slot (i.i.d.)
q[j] = the term count for v_j in query *q*.
$n_{\vec{q},L}$ = the number of queries *q'* of length L that correspond to the same count vector as q does, that is:

$$\vec{q}'=\vec{q}\,.$$

Hence we can estimate the probability of *q* being generated by the multinomial language model induced from *d* as:

$$\text{(Eq 15):}\quad P_{\theta_d}(\vec{q}) = n_{\vec{q},L}\prod_j (\theta_d[j])^{q[j]}$$

A few notes:

- The first term in equation 15 is document independent, so we never compute it in practice.
- We want $\theta_d[j] > 0$, because missing one query term shouldn't be equivalent to missing all query terms.
- We can justify having $\theta_d[j] > 0$ even if *d[j] = 0,* because *d* is a small sample from the source language model and therefore, we have no reason to expect that the actual probability of *v[j]* occurring is 0.

References

Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. SIGIR (1998). [pdf]

John Lafferty and Chengxiang Zhai. Probabilistic relevance models based on document and query generation. In Croft and Lafferty, eds., Language Modeling and Information Retrieval (2003). [pdf, ps]

ChengXiang Zhai and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. SIGIR (2001). [pdf, ps]

Lecture 9 INFO 630 / CS 674

The Language Modeling Approach to Information Retrieval Finger Exercises

Lecturer: Lillian Lee
Scribes: Vladimir Barash, Stephen Purpura, Shaomei Wu
September 25, 2007

Question 1. [*] The idea of language model relevance scoring is to calculate the probabilities of generating the given query under different language models, each of which is constructed from a document in the corpus. The higher probability a language model has to generate the query, the more relevant the associated document is considered to be. With such knowledge, given the *partial specification* of two multinomial language models M_1 and M_2, from two documents d_1 and d_2, respectively, calculate the relevance for them given the query "*September apple harvest festival*". (Note: The numbers in the table below represent the probability of generating the associated terms by M_1 and M_2. Not all the terms are included.)

Model M_1		Model M_2	
I	0.15	I	0.1
like	0.1	like	0.05
the	0.2	the	0.15
apple	0.02	apple	0.04
harvest	0.002	harvest	0.005
festival	0.005	festival	0.01
September	0.015	September	0.01

Sol:
Given the probability of each term generated by M1 and M2, we have:

q	September	apple	harvest	festival
M_1	0.015	0.02	0.002	0.005
M_2	0.01	0.04	0.005	0.01

$$P(\vec{q} \mid M_1) \underline{\underline{rank}}\ 0.015 \times 0.02 \times 0.002 \times 0.005 = 3e-9;$$
$$P(\vec{q} \mid M_2) \underline{\underline{rank}}\ 0.01 \times 0.04 \times 0.005 \times 0.01 = 2e-8;$$

Hence, given *q* and the multinomial language models, d_2 is more relevant than d_1.

Question 2. As given in Equation 15, under the multinomial language model, $P_{\theta_d}(\vec{q})$ denotes the relevance score of document *d* to given query *q*, evaluated as the probability of generating *q* from *d*'s language model; $\theta_d[j]$ represents the probability of choosing term v_j according to the LM induced from document *d*, and *q[j]* means the term count of v_j in query *q*.

$$\text{(Eq 15):}\quad P_{\theta_d}(\vec{q}) = (\text{number of } \vec{q}' \text{ of length } q[\bullet] \text{ s.t. } \vec{q}' = \vec{q}) \prod_j (\theta_d[j])^{q[j]}$$

[*] This question is derived from Example 12.1, Page 239 of *Introduction to Information Retrieval* (draft of November 13, 2007), by Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Cambridge University Press, 2008.

Given a simple approximation to $\theta_d[j]$ as $\frac{d[j]}{d[\bullet]}$, where *d[j]* is the term count for v_j in *d*, and $d[\bullet]$ is the length of *d* in terms, calculate the probability of generating query "apple harvest festival" from the language model inferred from the document *d* below (treat "apples" the same as "apple"):

> "Come celebrate Downtown Ithaca's 25th annual Apple Harvest Festival! Food, crafts, a ferris wheel, music... and did we mention apples? Apple cider, candy apples, caramel apples, apples by the bag, apple pie! orange you glad you like apples???"

Sol:

$d[\bullet] = 38$.

θ_d["apple"] = $\frac{d["apple"]}{d[\bullet]} = 8/38$;

θ_d ["harvest"] = $\frac{d["harvest"]}{d[\bullet]} = 1/38$;

θ_d ["festival"] = $\frac{d[" festival"]}{d[\bullet]} = 1/38$;

Hence $P_{\theta_d}("\text{apple harvest festival}") \underline{\underline{rank}} \frac{8}{38} \times \frac{1}{38} \times \frac{1}{38} = 0.0001458.$

Question 3: Compare and contrast the language modeling approach to information retrieval with the probabilistic (RSJ) and vector space modeling approaches to information retrieval.

Answer 3:

(1) Like the vector space model approach, the language modeling approach was initially empirically driven. However, its post-hoc theoretical explanation is, similarly to the probabilistic modeling approach based on stronger foundations than a pure vector space modeling approach.

(2) Although the probabilistic and the language modeling approaches can be interpreted as beginning with similar scoring functions, the language modeling approach we reviewed puts more emphasis on the importance of the query. One interpretation of the probabilistic modeling approach is the probability of a certain document attribute being exhibited by relevant documents given that a certain query was specified. In contrast, the language modeling approach can be interpreted as modeling the probability of a user providing a query given that they wish to retrieve documents generated from the same LM topic model.

Question 4: In [Ponte & Croft, SIGIR '98]'s view, users have a reasonable idea of terms that are likely to occur in documents of interest and will choose query terms that distinguish these documents from others in the collection. Doesn't this require a (potentially ridiculous) assumption that users have an idea of the distribution of terms in documents in the corpus?

Answer 4: Yes, the assumption is that the user will have an idea of the distribution of terms in the corpus. However, this assumption hasn't proven to be a significant limitation in empirical evaluation of performance. Apparently, like other models of retrieval, users must just have a sense for which words might distinguish their topic.

Despite this "reasonable sense", in later lectures we examine the effects of interactive query expansion. Ian Ruthven's "Re-examining the potential effectiveness of interactive query expansion" [SIGIR, 2003] shows that helping the user make effective decisions about the use of query terms is not easy.

Question 5: In the lecture notes, we use a Viterbi approximation to transition from equation 9 to equation 11 as follows:

$$(Eq\,9):\ \sum_t P(Q=q|T_Q=t)P(T_Q=t)P(T_D=t|D=d)$$

We cannot simplify equation 9 without further assumptions. In this case, we assume that estimation will be easier if we don't have to deal with all possible topics. So, using an assumption conceptually similar to a Viterbi approximation, we assume that there exists some *t*(d)* such that

$$(Eq\,10):\ P(T_D=t^*(d)|D=d)=1$$

Then equation 9 becomes:

$$(Eq\,11):\ P(Q=q|T_Q=t^*(d))P(T_Q=t^*(d))$$

as the term $P(T_D=t \mid D=d)$ is 0 for all *t* other than *t*(d)*, and *1 for t*(d*).

Explain why this assumption may not always be reasonable:

Answer 5 : In this case, the Virterbi approximation implies that the observed stream of words in the document is treated as an observed sequence of events. The document topic model is considered to be the "hidden cause" of the text in the document. The Viterbi approximation finds the most likely document topic model given the string of text. However, the "most likely" is not always the right answer. If there exist many possible topic models which could result in the observed stream of text, with relatively equal probability, then the system may return incorrect results frequently. Consider a hierarchical set of documents about social issues, with a sub-topic of health care, which includes discussions of issues about veterans health care, gang health care, health care for inner-city youth, and many other subtopics. If 99% of the documents are about topics other than health care and the documents about health care are unevenly split between the health care subtopics, how would the system distinguish between them? A user might get lucky and know the exact words which distinguish between the topics, but (unless they are an expert with intimate knowledge of the word distributions) it is more likely that the system will return results that roughly match their query,but will not be appropriately ranked. The user may then need to page through the results set to find documents which match their true interest. (As was mentioned in the answer to the last question, system designers need to be careful about their expectation of a user's ability to select appropriate query terms. [Ruthven, 2003])

Another problem with this assumption is that even if only one topic model predominates, saying that $P(T_D = t^*(d) \mid D = d) = 1$ may be quite far off the mark, causing the document in question to be erroneously highly ranked.

INFO 630 / CS 674 Deeper Thought Exercise

The Language Modeling Approach to Information Retrieval

Lecturer: Lillian Lee
Lecture 9: September 25, 2007
Scribes: Vladimir Barash, Stephen Purpura, Shaomei Wu

Question 3 of the Finger Exercise asked you to compare and contrast the language modeling approach to the vector space modeling approach in IR. This exercise will ask you to empirically compare the results given by these two approaches on a sample corpus, and more deeply explore the ranking space of different LM approaches.

Question: Given a sample corpus of 5 short documents (linked below), explore the effect of: a) using a stop word-removed, Porter-stemmed vocabulary; and b) using Dirchlet smoothing on document rankings for the queries ({cat, love} and {dog, love}), calculated using the LM approach. Then, compare the rankings you have calculated with VSM rankings. Does either approach perform definitively better on the corpus? Explain your reasoning.

Answer:

Calculations:

The LM score function for a document is:

$$(Eq\,1):\ P_{\theta_d}(\vec{q}) \overset{rank}{=} \prod_j (\theta_d[j])^{q[j]}$$

where:

$$(Eq\,2):\ \theta_d[j] = \frac{d[j]}{d[.]}$$

without Dirchlet smoothing, and:

$$(Eq\,3):\ \theta_d[j] = \frac{d[j] + \mu \frac{C[j]}{C[.]}}{d[.] + \mu}$$

with Dirchlet smoothing. **C**[.] is the sum of term frequencies over all terms, and **d**[.] is the document length in words, including repeats. We use a μ of 2,000.

Rankings:

{cat, love}

Full vocabulary:
Document # no-smooth yes-smooth
1 0 0.000070202
2 0 0.000077433
3 0 0.000067883
4 0 0.000068284
5 0 0.000069661

rank-no-Dirichlet-smooth-LM: **d1=d2..d5**
rank-Dirichlet-smooth-LM: **d2 > d1 > d5 > d4 > d3**
rank-L2-VSM: **d2 > d5 > d1, d3, d4**
rank-pivoted-VSM: **d5 > d2 > d1, d3, d4**

Porter Stemmed vocabulary:
Document # no-smooth yes-smooth
1 0 0.001667827
2 0 0.001699982
3 0 0.001649676
4 0 0.001644524
5 0.00167 0.001667936

rank-no-Dirichlet-smooth-LM: **d5 > d1...d4**
rank-Dirichlet-smooth-LM: **d2 > d5 > d1 > d3 > d4**
rank-L2-VSM: **d2 > d5 > d1 > d3 > d4**
rank-pivoted-VSM: **d5 > d2 > d1 > d3 > d4**

{dog, love}

Full vocabulary:
Document # no-smooth yes-smooth
1 0 0.000070202
2 0 0.000077433
3 0 0.000067883
4 0 0.000072973
5 0 0.000065324

rank-no-Dirichlet-smooth-LM: **d1=d2...d5**
rank-Dirichlet-smooth-LM: **d2 > d4 > d1 > d3 > d5**
rank-L2-VSM: **d2 > d4 > d5 > d1, d3**
rank-pivoted-VSM: **d2 > d4 > d5 > d1, d3**

Porter Stemmed vocabulary:
Document # no-smooth yes-smooth
1 0 0.001450465
2 0.06250 0.001497897
3 0 0.001444722
4 0 0.0014389587
5 0.00139 0.0014569805

rank-no-Dirichlet-smooth-LM: **d2 > d5 > d1,d3,d4**
rank-Dirichlet-smooth-LM: **d2 > d5 > d1 > d3 > d4**
rank-L2-VSM: **d2 > d5 > d3 > d1, d4**
rank-pivoted-VSM: **d5 > d2 > d3 > d1, d4**

Observations:
First, it is plain that the no-smooth rankings are much worse than the Dirichlet smoothed rankings. The

no-smoothing approach suffers from the "sparse data" problem: in a small corpus of small documents such as this one, the probability of any given document having all of the terms in a given query is low. But if a document lacks any of the query terms, the θ_j for that term = 0. Then:

$\theta_j^{q[j]} = 0$

which zeros out that document's final score with respect to q. In sum, no-smooth LM rankings are only potentially good for differentiating between documents in a corpus of large documents, where the probability of any relevant given document having all of the query terms is high. We do not expect such corpora to be realistic.

Second, the Dirichlet-smoothing LM approach seems to greatly prefer short documents to long documents. On {cat, love} with the full vocabulary, LM with Dirichlet smoothing places d1, a short document, in second place, over both of the VSM approaches, which place it 3rd. This bias does seem to be consistent for both queries, across the full and Porter-stemmed vocabularies (especially on {dog, love} with the full vocabulary, where LM with Dirichlet smoothing places the longest document dead last, even though it contains one of the search terms!). The value of μ may play a role in creating this bias: for instance, if μ = 1, the long document d5 comes out ahead of d3. Perhaps a smaller value of μ would produce a ranking that is less biased in favor of short documents.

Thirdly, the Dirchlet-smoothing LM approach does not prefer short documents over long documents quite so much in the case of a stop-word-removed vocabulary. This is probably due to the nature of the Dirchlet-smoothing function. The denominator of this function is **d**[.] + a constant (note the similarity to L_1 normalization). **d**[.] values are smaller for a stop-word-removed vocabulary than for the original vocabulary, especially in the case of large documents, as large documents tend to contain more stop words. So, it is reasonable to assume that the removal of stop words raises all document scores (as we see in the data) and that it raises the scores of large documents more than the scores of small documents (as we also see in the data).

In conclusion, we cannot definitively say that the LM approach is better than the VSM approach - LM shows comparable, and in some cases, worse rankings to VSM on the test corpus. It is clear, however, that Dirichlet or some other means of smoothing is essential for calculating rankings using LM. Furthermore, the test corpus seems to suggest that LM is biased to rank short documents over long ones; if this bias holds up for larger corpora, it can and should be compensated for (perhaps by an normalization factor, or by regulating the degree of smoothing?).

References:

Jin, Rong. "Language Modeling Approaches for Information Retrieval," online at: http://www.cse.msu.edu/~cse484/lectures/lang_model.ppt

See http://docs.google.com/Doc?id=dcpkz9gb_42wmz5b5 for the full texts, processing instructions, and raw statistics about the text.

For calculations of L2 and pivoted-based normalization rankings, see http://docs.google.com/Doc?id=dcpkz9gb_94csg6xs

For a spreadsheet of the full vocabulary document matrix and statistics, see: http://spreadsheets.google.com/pub?key=pflK40UHlXXmSucfR-fq7TA

For a spreadsheet of the Porter stemmed vocabulary document matrix and statistics, see: http://spreadsheets.google.com/pub?key=pflK40UHlXXmWPjYGVQNoZQ