CS 630 Lecture 4: 02/07/2006
Lecturer: Lillian Lee Scribes: Peter Babinski, David Lin

Probabilistic Retrieval

I. Naïve Beginnings

a. Motivations

After a few days of talking about the vector space model and convincing ourselves that such an ad-hoc approach works, it seems strange to take a step back and decide to start over. However, it is clear that in defining the vector space model, we made a few potentially controversial judgments. For example, why does a bag of words feature vector make sense anymore than some other feature vector? Why not use feature vectors that tracks counts of the ten most common words, the lengths of those words, the number of paragraphs, and the number of idioms used in a document? Why do we use this thing called term frequency, TF, and inverse document frequency, IDF, in the similarity function we use to rank the documents by their relation to a query? These are the types of questions that lead researchers to consider a probabilistic model.

Credits:

The researchers that developed the probabilistic model for information retrieval include Robertson, Spärk Jones, and Walker, among many others

Goals:

- Gain a better theoretical understanding of information retrieval.
- Obtain state-of-the-art performance.

b. "False Start": A Probabilistic Model without Variation?

Note: For the following discussion, unless otherwise stated, we are assuming that *q*, the query, is fixed.

We start by mentioning that the fundamental principle of our model is to rank documents by the probability that they are relevant, noting that this idea of relevance is very fuzzy.

In other words, or rather without using English words, we want to compute the following:

(1) P(relevant | document)

Only a few seconds into our discussion, we see a glaring problem. Where are the random variables in this equation?

For those who do not have much experience with random variables, we mention the following rules:

- Use capital letters for random variables
- Use lower case letters for values of a random variable (r.v.)

Ex. We can declare r.v. X to be the value of a fair coin and can compute the probability P(X = x), where $x \in \{Heads, Tails\}$.

In response to our lack of random variables, we can assign a random variable to the relevance, R, and the document, D, and rewrite equation **(1)** as the following:

(2) P(R = r | D = d)

Now that we have random variables, we have to decide what possible values they can take. For example, we can let R take on the values $r \in \{yes, no\}$ or $r \in \{0,.1,.2,\ldots,.9,1\}$. Whether we use the former or the latter, there is still no variation, because in the former case, we already know whether a document is either relevant or not. In the latter case, there is still no difference, as we already know if a document has a relevance rating of .5 or 0.

We also need to define what possible values D can take. We can either use *d* to represent a document in a given corpus, C, or as a document from the space of all possible documents. We also need to take care that P(D = d) > 0 for all documents d, regardless of which option we choose.

Regardless of our choices here, we still have not managed to introduce variance into our probability formula, **(2)**.

Let's consider a few of the possible sources of variance that we could add:

- Variation in a specific user's judgment of whether a document is relevant or not. The user may change his mind about whether a document is relevant based on when you ask him. For example, the user may use the same term to mean different things at different times.
- Variation over different users [Maron and Kuhns (1960)]. Each user may have a different opinion on whether a document is relevant or not to a given query. The main concern is how we would characterize this variation without some model of human preferences and beliefs.
- Variation because of the document representation [Robertson and Spärck Jones (1976)]. Some documents with a certain representation are relevant, while some are not. This idea stems from the fact that if we use a document representation that incorporates terms without the various language constructs or context they are found in, then some documents with a certain feature vector may be more relevant to a query than others with the same feature vector. For example, a document containing the phrase "the White House is" has a different topic than a document that only contains the phrase "the house is white".

For lack of a better choice, and since using reduced document representations is fairly standard practice in IR, we will try the route of introducing variation in relevance due to the document representation. Still, is there another source of variation we could introduce that we aren't considering?

II. Formulation

Note: For the following discussion, unless otherwise stated, we are assuming that *q*, the query, is fixed.

a. Terms and Definitions

Previously, we decided that we would introduce variation into our probabilistic retrieval model by allowing a document with a certain representation to be relevant where another document with the same representation is not relevant. This feature of our model motivates our ensuing discussion on "binning" the documents into classes.

We will represent each document with a set of attributes:

$\vec{A} = (A_1, ..., A_m)^T$ where A_j is an r.v. for attribute j's value

Therefore, for each document *d*, there is an associated vector:

$\vec{a}(d) = (a_1(d), ..., a_m(d))^T$

Ex. $a_j(d)$ = the number of times $v^{(j)}$ occurs in d or $a_j(d) = \begin{cases} 1 & \text{if } v^{(j)} \text{ in d} \\ 0 & \text{else} \end{cases}$ [or even $a_{m+1}(d) = \text{length}(d)$]

We can already see that there may be some problems with ambiguity in this representation.

Ex. Take the documents "white house" = $d^{(1)}$ and "house white" = $d^{(2)}$. If the query is asking for documents dealing with U.S. politics, then only document (1) is relevant, even though $\vec{a}(d(1)) = \vec{a}(d(2))$.

Using our attribute vectors, we can rewrite our probability formula from equation **(2)** as the following:

(3) $P(R = y \mid \vec{A} = \vec{a}(d))$

Some immediate concerns arise however. Wasn't our choice of features for this representation of the document just as arbitrary as the bag of words representation we decided upon for the VSM (vector space model)?

b. Making Stuff Computable

We next examine if we can "derive" the VSM from our probabilistic retrieval model as some special case. In order for this result, we will need to eliminate or compute the terms that we have left from equation **(3)**.

We observe that the event that $\vec{A} = \vec{a}(d)$ is likely very rare unless we reduce our attribute space such that more documents are in each "bin". There is also a lack of relevance statistics (recall that we have no relevance-labeled documents). Therefore, we will try to use Bayes Rule to see if we can obtain a probability formula that we actually can compute. Recall that Bayes Rule is the following:

(4) $$P(A \mid B) = \frac{P(B \mid A)\, P(A)}{P(B)}$$

Applying this formula to **(3)**, we obtain:

(5) $$P(R = y \mid \vec{A} = \vec{a}(d)) = \frac{P(\vec{A} = \vec{a}(d) \mid R = y)\, P(R = y)}{P(\vec{A} = \vec{a}(d))}$$

And under ranking equivalence [ranking equivalence means that we can eliminate terms that do not depend on the documents since they will be ignored when we rank documents by this function] for documents, **(5)** becomes:

(6) $$\frac{\mathrm{P}(\vec{A}=\vec{a}(d)\,|\,\mathrm{R}=\mathrm{y})\,\mathrm{P}(\mathrm{R}=\mathrm{y})}{\mathrm{P}(\vec{A}=\vec{a}(d))} \xrightarrow{\text{rank}} \frac{\mathrm{P}(\vec{A}=\vec{a}(d)\,|\,\mathrm{R}=\mathrm{y})}{\mathrm{P}(\vec{A}=\vec{a}(d))}$$

The next step requires us to make some broad assumptions that don't seem possible to justify. We need to break down this term some more, so we try to assume conditional independence between the elements of $\vec{A}$ and R and independence between elements of $\vec{A}$ in general.

As Cooper suggests, this modeling assumption is a major inconsistency in the probabilistic retrieval model. He in turn supports his statement with an example where assuming both independence and conditional independence gives a logical inconsistency.

In response, we can instead rely on a sort of weak link dependence assumption to obtain the following:

(7) $$\frac{\mathrm{P}(\vec{A}=\vec{a}(d)\,|\,\mathrm{R}=\mathrm{y})}{\mathrm{P}(\vec{A}=\vec{a}(d))} \xrightarrow{\text{w.l.d.a}} \frac{\prod_{\mathrm{j}} \mathrm{P}(\vec{A}_j=\vec{a}_j(d)\,|\,\mathrm{R}=\mathrm{y})^{\,\alpha}}{\prod_{\mathrm{j}} \mathrm{P}(\vec{A}_j=\vec{a}_j(d))} \quad \alpha>0, \alpha\neq 1$$

In turn, the α disappears when ranking documents in any case. Note some effects of this choice. There are some attractive features to this decomposition.

Ex. Say we have $\mathrm{d}^{(1)}$ = "White" and $\mathrm{d}^{(2)}$ = "House".

Let A_1 = does "white" appear? A_2 = does "house" appear?
$\vec{a}(d)^{(1)} = (1, 0)$ $\vec{a}(d)^{(2)} = (0, 1)$

We would find that the probability of "White House" appearing is computed by this equation, $\frac{\#(\text{"}White\text{"})}{\#docs} \times \frac{\#(\text{"}House\text{"})}{\#docs}$. Although no document contains "White House", this probability we computed will be greater than zero, which is reasonable.

We still have problems with these terms conditioned on the relevance of the document, which we still have no way to compute. What other information can we take advantage of? We still have our query, q, which may be our only clue to the value of R.

Let $q_{\mathrm{j}} = \begin{cases} 1 & \text{if } \mathrm{v}^{(j)} \text{ in } q \\ 0 & \text{else} \end{cases}$

Intuitively, we can take equation **(7)** and split the terms in the product sum by whether they occur in the query or not (whether q_j is equal to 1 or 0). Thus we have the following equation:

(8) $$\prod_{j:\, q_j=1} \frac{P(\vec{A}_j = \vec{a}_j(d) \mid R = y)}{P(\vec{A}_j = \vec{a}_j(d))} \times \prod_{j:\, q_j=0} \frac{P(\vec{A}_j = \vec{a}_j(d) \mid R = y)}{P(\vec{A}_j = \vec{a}_j(d))}$$

Once again, we make a simplifying assumption. What if the terms not in the query (q_j = 0) have the same distribution over relevant documents as over all other documents? If we can afford to make this assumption, then the second term in **(8)** will disappear entirely under equivalence ranking because $P(\vec{A}_j = \vec{a}_j(d) \mid R = y) = P(\vec{A}_j = \vec{a}_j)$ if $q_j = 0$.

Therefore, we find ourselves at the following simpler, but still difficult to compute function:

(9) $$\prod_{j:\, q_j=1} \frac{P(\vec{A}_j = \vec{a}_j(d) \mid R = y)}{P(\vec{A}_j = \vec{a}_j(d))}$$

In conclusion, we see that our approach is indeed helping us to simplify the term. However, we still are not at a point where we can estimate the part of the equation relating to R. We will need to continue simplifying **(9)** to reach a point where we can use some (any?) estimation to deal with our lack of relevance information.

III. Related Questions

Let us examine how we might use our probabilistic model if we had some training data with relevance information labeled for a fixed query.

Say we have m = 5 features, listed below:

Element of $\vec{a}_j$	1	2	3	4	5
Feature	"computer" in doc	"science" in doc	"research" in doc	"program" in doc	"fishing" in doc

Each element of $\vec{a}_j$ is equal to the number of occurrences of a feature if that feature is present in the document. In order to calculate probabilities, we will need to use counts over our corpus, which is only the four documents below. We can estimate the probability of any feature in our feature vector by summing the occurrences of that feature value among the documents (restricted to relevant ones in the case of R = y) and dividing by the number of documents.

Given Training Documents:
$d^{(0)}$ = "some new research in computer architecture", R = y
$d^{(1)}$ = "computer have yet to change age-old methods of fishing", R = n
$d^{(2)}$ = "computer aided research in lure dynamics", R = n
$d^{(3)}$ = "computer science has created a plethora of new program", R = y

A) Compute the attribute vectors for the documents above, and then compute P(R = y), $P(\vec{A}_j = 1)$, and $P(\vec{A}_j = 1 \mid R = y)\ \forall j$.

Answer:

$a(d^{(0)}) = (1, 0, 1, 0, 0)^T$
$a(d^{(1)}) = (1, 0, 0, 0, 1)^T$
$a(d^{(2)}) = (1, 0, 1, 0, 0)^T$
$a(d^{(3)}) = (1, 1, 0, 1, 0)^T$

$$P(R = y) = \frac{\#relevant\ docs}{\#docs} = \frac{2}{4} = .5 \quad P(R = n) = 1 - P(R = y) = .5$$

Recall that the probability $P(\vec{A}_j = \vec{a}_j(d))$ for a term $v^{(j)}$ is computed by the following:

$$P(\vec{A}_j = \vec{a}_j(d)) = \frac{\#docs\ d'\ s.t.\ \vec{a}_j(d') = \vec{a}_j(d)}{\#docs}$$

And for the probability $P(\vec{A}_j = \vec{a}_j(d) \mid R = y)$ for a term $v^{(j)}$ is computed by the following:

$$P(\vec{A}j = \vec{a}j(d) \mid R = y) = \frac{\#docs\ d'\ s.t.\ d'\ is\ relevant\ and\ \vec{a}_j(d') = \vec{a}_j(d)}{\#relevant\ docs}$$

$P(\vec{A}_1 = 1) = 1$
$P(\vec{A}_2 = 1) = .25$
$P(\vec{A}_3 = 1) = .5$
$P(\vec{A}_4 = 1) = .25$
$P(\vec{A}_5 = 1) = .25$
$P(\vec{A}_1 = 1 \mid R = y) = 1$
$P(\vec{A}_2 = 1 \mid R = y) = .5$
$P(\vec{A}_3 = 1 \mid R = y) = .5$
$P(\vec{A}_4 = 1 \mid R = y) = .5$
$P(\vec{A}_5 = 1 \mid R = y) = 0$

Conversely, we also know the probabilities of each attribute vector term being 0.

B) Immediately we noticed one of the problems that we considered during our discussion. Since the word "fishing" was never found among our relevant documents, the probability $P(\vec{A}_5 = 1 | R = y) = 0$ became zero. However, we would like to be able to account for cases where we haven't seen a word in our corpus. This situation may occur if we are obtaining new documents that are not in our training corpus and would like to immediately classify them as relevant or not relevant to a fixed query. If we want to better estimate counts for our model, how could we modify the function we use to estimate probabilities? Recall we used a formula like this:

$$P(\vec{A}_j = \vec{a}_j(d)) = \frac{\#docs\, d' s.t. \vec{a}_j(d') = \vec{a}_j(d)}{\#docs}$$

Answer:

There are several ways in which we could modify how we estimate probabilities from our counts. The simplest method would be to add one to all our counts s.t. the new equation is:

$$P(\vec{A}_j = \vec{a}_j(d)) = \frac{[\#docs\, d' s.t. \vec{a}_j(d') = \vec{a}_j(d)] + 1}{\#docs}$$

m is the number of features in our feature vector.

Another slightly more complicated method is to estimate the counts of features that we haven't seen by using the counts of features that we've only seen in one document. We could redistribute some of the probability mass from all words seen once to those words not yet seen in our feature set.

A still smarter idea would be to consider any low number of counts (less than 5 for example) to be unreliable and redistribute these counts to those counts that are zero.

Regardless, these "smoothing" methods are a heady topic on their own. E.g. Chen and Goodman '98.

C) What if we decided to incorporate a greater detail of resolution in our relevance labels? For example, we could let $R \in \{0,.1,.2,\ldots,.9,1\}$ instead of just "y" or "n". What is one problem that such a change would cause when we tried to actually estimate the probabilities for such a model?

Answer:

By now you likely know that adding additional classes would only decrease the amount of documents of each relevance level that we could use to estimate the probabilities, which would likely hurt the performance of this estimated model. As an open ended question, consider what would happen if we magically had enough data to properly train our model? Knowing that we intend to rank the documents through some scoring scheme anyway, would a better relevance resolution scheme necessarily help us?