University of Illinois at Urbana-Champaign

Midterm Examination

CS410 Introduction to Text Information Systems
Professor ChengXiang Zhai
TA: Azadeh Shakery

Time: 2:00–3:15pm, Mar. 14, 2007
Place: Room 1105, Siebel Center

Name:_____

NetID:_____

1. **[10 points] Evaluation**

   (a) **[5/10 points]** Suppose we have a topic (i.e., a query) with a total of 5 relevant documents in the whole collection. A system has retrieved 6 documents whose relevance status is

   $$[+, +, -, -, -, +]$$

   in the order of ranking. A "+" (or "-") indicates that the corresponding document is relevant (or non-relevant). For example, the first two documents are relevant, while the third is non-relevant, etc. Compute the precision, recall, and the (non-interpolated) mean average precision for this result.

   Solution:
   precision = $\frac{3}{6} = 0.5$
   recall = $\frac{3}{5} = 0.6$
   MAP = $(\frac{1}{1} + \frac{2}{2} + \frac{3}{6})/5 = 0.5$

   (b) **[3/10 points]** Briefly explain why precision at k (e.g., $k = 10$) documents is not as good as the (mean) average precision for comparing two retrieval systems in terms of their ranking accuracy.

   Solution:
   Precision at k documents is not sensitive to the order of the documents as far as they are among the top k.
   MAP is sensitive to the entire ranking and also contains recall-oriented aspects.

   (c) **[2/10 points]** Does precision at k (e.g., $k = 10$) documents have any advantage over the (mean) average precision as a measure for evaluating retrieval performance?

   Solution:
   Yes, in cases where the focus is only on the top k documents. For example in Web search, users usually look at the first few results.

2. **[25 points] Bayes Rule**

Author H and author T are co-authoring a paper in the following way:

(a) Each word is written independently.

(b) When writing a word, they would first toss a coin to decide who will write the word. The coin is known to show up as HEAD 80% of the time. If the coin shows up as HEAD, then author H would write the word, otherwise, author T would write the word.

(c) If it is author H's turn to write, he would "write" the word by simply drawing a word according to word distribution θ_H. Similarly, if it is author T's turn to write, he would "write" the word by drawing a word according to word distribution θ_T.

Suppose the two distributions θ_H and θ_T are defined as follows:

Word w	$p(w\|\theta_H)$	$p(w\|\theta_T)$
the	0.3	0.3
computer	0.1	0.2
data	0.1	0.1
baseball	0.2	0.1
game	0.2	0.1
...	...	...

(a) **[6/25 points]** What is the probability that they would write "the" as the first word of the paper? Show your calculation.

Solution:
$p(w = \text{“}the'') = p(w = \text{“}the''|\Theta_H)p(\Theta_H) + p(w = \text{“}the''|\Theta_T)p(\Theta_T) = 0.8 \times 0.3 + 0.2 \times 0.3 = 0.3$

(b) **[4/25 points]** What is the probability that they would write "the" as the second word of the paper?

Solution:
Since each word is drawn independently, the probability of writing "the" as the second word is the same as the probability of writing "the" as the first word, i.e. 0.3.

(c) **[6/25 points]** Suppose we observe that the first word they wrote is "data", what is the probability that this word was written by author H (i.e., $p(\theta_H|w ='' data'')$)? Show your calculation.

Solution:
$p(\Theta_H|w = \text{“}data'') = \frac{p(w=\text{“}data''|\Theta_H)p(\Theta_H)}{p(w=\text{“}data''|\Theta_H)p(\Theta_H)+p(w=\text{“}data|\Theta_T)p(\Theta_T)} = \frac{0.1\times0.8}{0.1\times0.8+0.1\times0.2} = 0.8$

(d) **[4/25 points]** Imagine that we observe a very long paper written by them (e.g., with more than 10,000 words). Among the 5 words shown in the table above (i.e., "the", "computer", "data", "baseball", "game"), which one would you expect to occur least frequently in the paper? Briefly explain why.

Solution:
"data" would occur least frequently, since $p(w = \text{“}data'')$ is the smallest.

(e) **[5/25 points]** Suppose we don't know θ_H, but observed a paper D known to be written *solely* by author H. That is, the coin somehow always showed up as HEAD when they wrote the paper. Suppose $D =$"the computer data the computer game the computer data game" and we would like to use the maximum likelihood estimator to estimate θ_H. Fill in the values for the following estimated probabilities:

Word w	$p(w\|\theta_H)$
the	0.3
computer	0.3
data	0.2
baseball	0
game	0.2

3. **[10 points] Zipf's law.** Assume that the frequency distribution of words in a collection of documents C roughly follows the Zipf's law $r * p(w_r|C) = 0.1$, where $r = 1, 2, 3, ...$ is the rank of a word in the descending order of frequency. w_r is the word at rank r, and $p(w_r|C)$ is the probability (frequency) of word w_r in the collection. What is the probability of the most frequent word in the collection? What is the probability of the second most frequent word in the collection?

Solution:

From Zipf's law: $p(w_r|C) = \frac{0.1}{r}$

Probability of the most frequent word: $p(w_1|C) = \frac{0.1}{1} = 0.1$

Probability of the second most frequent word: $p(w_2|C) = \frac{0.1}{2} = 0.05$

4. **[20 points] Vector Space Model**

Consider the following "TF-IDF" retrieval formula:

$$score(D, Q) = \sum_{w \in Q \cap D} (c(w, D) + IDF(w))/IDF(w)$$

where $c(w, D)$ is the raw count of word w in document D, and $IDF(w)$ is the IDF of word w.

(a) **[8 points]** Point out at least two reasons why the formula above is unlikely to perform well empirically.

Solution:

- Wrong usage of IDF: Larger IDF leads to lower weight
- No document length normalization
- No TF normalization
- No query term frequency (qtf)

(b) **[6 points]** How would IDF(w) change (i.e., increase, decrease or stay the same) in each of the following cases: (1) adding the word w to a document; (2) make each document twice as long as its original length by concatenating the document with itself.

Solution:
(1) If the document already contains w, $IDF(w)$ will not change, otherwise it will decrease, since the number of documents containing w is increased.
(2) $IDF(w)$ does not change, since the number of documents containing w does not change.

(c) **[6 points]** Briefly sketch how you may use the vector space retrieval method (e.g., pivoted normalization retrieval function, Rocchio feedback method) to design a spam filter. Let $S = \{s_1, ..., s_n\}$ be a set of n known spam messages, and $R = \{r_1, ..., r_m\}$ a set of m regular email messages. Describe how you can build a spam filter based on S and R such that it can process any new email message e and decide whether it is a spam.

Possible Solution:

(1) Represent each email message as a term vector

(2) Calculate the centroids of S and R, s_c and r_c

(3) Given a new email message e, measure the similarity between e and the centroids s_c and r_c. e is reported as spam if it is more similar to s_c, or not a spam if it is more similar to r_c.

(4) Once we get the feedback from user whether e is a spam or not, we add e to S or R respectively and update the centroids in step (2) according to the Rocchio feedback formula.

5. **[25 points] Dirichlet prior smoothing and retrieval**

Suppose we have a document collection with an extremely small vocabulary with only 6 words w_1, ..., w_6. The following table shows the estimated reference language model $p(w|REF)$ using the whole collection of documents (2nd column) and the word counts for document d_1 (3rd column) and d_2 (4th column), where $c(w, d_i)$ is the count of word w in document d_i. Let $Q = w_1 w_2$ be a query.

Word	$p(w\|REF)$	$c(w, d_1)$	$c(w, d_2)$
w_1	0.8	2	7
w_2	0.1	3	1
w_3	0.025	2	1
w_4	0.025	2	1
w_5	0.025	1	0
w_6	0.025	0	0
SUM	1.0	10	10

(a) **[10 points]** Suppose we do *not* smooth the language model for d_1 and d_2. Compute the likelihood of the query for both d_1 and d_2, i.e., $p(Q|d_1)$ and $p(Q|d_2)$. (Do not compute the *log*-likelihood.) Show your calculations. Which document would be ranked higher?

Solution:
$p(Q|d_1) = \frac{2}{10} \times \frac{3}{10} = 0.06$
$p(Q|d_2) = \frac{7}{10} \times \frac{1}{10} = 0.07$
d_2 would be ranked higher

(b) **[10 points]** Suppose we now smooth the language model for d_1 and d_2 using Dirichlet prior smoothing method with $\mu = 10$. Recompute the likelihood of the query for both d_1 and d_2, i.e., $p(Q|d_1)$ and $p(Q|d_2)$. (Do not compute the *log*-likelihood.) Show your calculations. Which document would be ranked higher this time?

Solution:
$p(Q|d_1) = (\frac{10}{10+10} \times \frac{2}{10} + \frac{10}{10+10} \times \frac{8}{10}) \times (\frac{10}{10+10} \times \frac{3}{10} + \frac{10}{10+10} \times \frac{1}{10}) = 0.1$
$p(Q|d_2) = (\frac{10}{10+10} \times \frac{7}{10} + \frac{10}{10+10} \times \frac{8}{10}) \times (\frac{10}{10+10} \times \frac{1}{10} + \frac{10}{10+10} \times \frac{1}{10}) = 0.075$
d_1 would be ranked higher

(c) **[5 points]** Intuitively, which document do you think should be ranked higher? d_1 or d_2? Why?

Solution:
d_1 should be ranked higher, since it contains a larger number of the more informative word w_2. From the estimated reference language model, it is clear that w_1 is a popular word and w_2 is more discriminative.

6. **[10 points] Compression and Information Theory**

(a) **[5 points]** Write down the gamma code for the integer 11.

Solution:
1110011

(b) **[5 points] Efficient Computation of smoothed language models** When a smoothed language model is involved in a problem, the computation generally involves a sum over all the words in the vocabulary due to the fact that smoothing assigns non-zero probability to all words. However, it is often possible to rewrite a formula so that the computation can be done much more efficiently than taking a brute-force sum over all words, as in the case of the query-likelihood retrieval formula.

Assume that a smoothed unigram language model for document D is given by

$$p(w|D) = \begin{cases} p_s(w|D) & \text{if word } w \text{ is seen} \\ \alpha_D\, p(w|C) & \text{otherwise} \end{cases}$$

where $p_s(w|D)$ is the smoothed probability of a word seen in the document, $p(w|C)$ is the collection language model, and α_d is a coefficient controlling the probability mass assigned to unseen words, so that all probabilities sum to one.

Show that the following formula correctly computes the entropy of any smoothed document language model in this way. (Note that this formula has two parts – one part is a sum over all words in D and the other is the collection model entropy which can be pre-computed.)

$$H(\theta_D) = \sum_{w \in D} [\alpha_D p(w|C) log(\alpha_D p(w|C)) - (p_s(w|\theta_D) \log p_s(w|\theta_D))] - \alpha_D \log \alpha_D - \alpha_D \sum_{w \in V} p(w|C) \log p(w|C)$$

where V is the set of all the words in the vocabulary.

Solution:

$$\begin{aligned}
H(\Theta_D) &= -\sum_{w \in V} p(w|\Theta_D) \log p(w|\Theta_D) \\
&= -\sum_{w \in D} p_s(w|D) \log p_s(w|D) - \sum_{w \notin D} \alpha_D p(w|C) \log(\alpha_D p(w|C)) \\
&= -\sum_{w \in D} p_s(w|D) \log p_s(w|D) - \sum_{w \in V} \alpha_D p(w|C) \log(\alpha_D p(w|C)) + \sum_{w \in D} \alpha_D p(w|C) \log(\alpha_D p(w|C)) \\
&= \sum_{w \in D} [\alpha_D p(w|C) \log(\alpha_D p(w|C)) - p_s(w|D) \log p_s(w|D)] - \sum_{w \in V} \alpha_D p(w|C) \log(\alpha_D p(w|C)) \\
&= \sum_{w \in D} [\alpha_D p(w|C) \log(\alpha_D p(w|C)) - p_s(w|D) \log p_s(w|D)] \\
&\qquad - \sum_{w \in V} \alpha_D p(w|C) \log \alpha_D - \sum_{w \in V} \alpha_D p(w|C) \log p(w|C) \\
&= \sum_{w \in D} [\alpha_D p(w|C) \log(\alpha_D p(w|C)) - p_s(w|D) \log p_s(w|D)] - \alpha_D \log \alpha_D - \alpha_D \sum_{w \in V} p(w|C) \log p(w|C)
\end{aligned}$$

scratch