Discussion

Pre-workshop

• Review the BM25 model (from WSTA) and the "Jelinek-Mercer" language model (from KT/WSTA). Identify the variables that go into each similarity model, and the parameters (and what they control).

• What is probability “smoothing”? Why is it important?

• Why the web called a “graph”? What properties of the graph can we leverage to produce better IR results?

• What is “PageRank” based on? How is it calculated? And “HiTS”?

Workshop

1. What is a “language model”?
   (a) What does an “n-gram language model” refer to?
   (b) What is the significance of \( \hat{P}(q \mid d) = P(q \mid M_d) \)? How would we go about estimating \( P(q \mid M_d) \)?

2. What are the following smoothing methods?
   (a) “Uniform prior” (a.k.a. “Laplacian” or “add-one”)  
   (b) “Linear interpolation wrt to a background universe” (a.k.a. “back-off”)  
   (c) “Dirichlet” (a.k.a. “categorical conjugate prior”)

3. Consider a (toy) collection, with 3 documents \( D_1, D_2, D_3 \). \( D_1 \) has a link to \( D_2 \), \( D_2 \) has a link to \( D_3 \), and \( D_3 \) has a link to \( D_2 \).
   (a) Find the weights of the three documents, according to PageRank, with \( \alpha = 0.5 \). (Note that even with such a small collection, doing this is far easier with a statistical package like R or Matlab.)
   (b) Calculate 3 iterations of the HiTS algorithm, to find the hub and authority scores of the 3 documents.

Post-workshop

• In an n-gram language model, what do you think is a suitable value for \( n \) when representing common IR queries? What criteria go into choosing a “suitable” value?

• You may have seen some instances of the three smoothing methods above in this subject or KT. Try to identify some instances where they commonly arise (e.g. Naive Bayes).
Programming

Pre-workshop

- NLTK has some support for building n-gram language models (as `ConditionalFreqDist`). Familiarise yourself with some of the relevant functions discussed in http://www.nltk.org/book/ch02.html#conditional_frequency_distribution_and_index_term, or more technically, http://www.nltk.org/_modules/nltk/model/ngram.html.

- Choose a suitable matrix decomposition package (for example, `numpy` has some eigenvector routines in its `linalg` package).

Workshop

1. Build a 2-gram language model over one or more texts, and use it to find the (log-scaled) conditional probability of the query given the model’s document(s).

2. Numerically find the matrix solution to the PageRank problem above.

Post-workshop

- Build a couple of more language models, and compare the probability according to a different set of documents (e.g. the TREC blog data vs. *Moby Dick*, or one TREC blog document vs. another). Observe how you could use this to build a ranked query IR engine.

- Draw a graph made of a collection of 10 documents, and numerically calculate PageRank again. How much longer does the algorithm take to find the solution (and is there a solution?)? What problems will arise as you scale the graph up to the size of the World Wide Web?