Language Models and Structured Documents

Paul Ogilvie
Carnegie Mellon University

Motivation: Language Models

- Successful in IR
  - Ad-hoc database retrieval (C. Zhai and J. Lafferty)
  - Distributed retrieval (Si Luo et al)
  - Named-page finding (K. Collins-Thompson et al)
- Established techniques for estimating distributions from small amounts of text
- Easily extended to model document structure

Language Models in Named-Page Finding

- Generative language model

$$P(Q|\theta_D) = \prod_{w \in (q_1, q_2, \ldots, q_n)} P(w|\theta_D)$$

- Language models created from different document representations

Full text	Title	In-link	Image alt text
Meta tags	URL	Fonts and Headers	

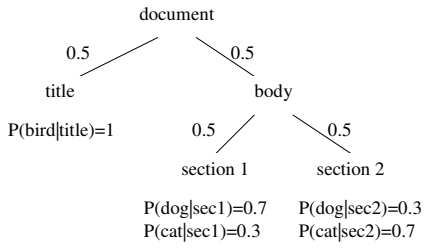
- Estimate the page name model by a linear interpolation of the other models

$$P(w|\theta_D) = \sum_i \lambda_i P(w|\theta_{D,i})$$

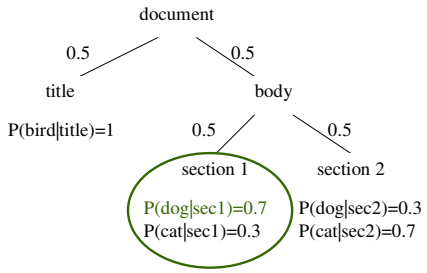
Proposed Model for Structured Documents

- Model document structure as a "parse tree"
- Estimate a language model for each node
  - Leaf nodes estimated directly from text
  - Use a linear interpolation estimate parent's model
- We know the structure, so we don't need to estimate the probability of rules
- But we do need to estimate the linear interpolation parameters…

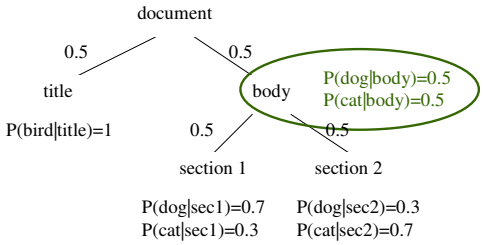
A Toy Document



Query "dog"



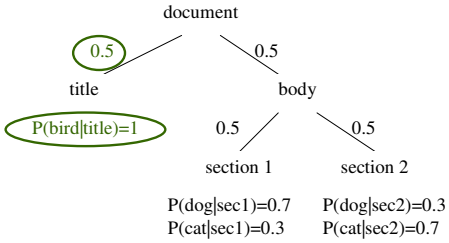
Query “dog cat”



Structured Queries

- a AND b = P(a)*P(b)
- a OR b = P(a) + P(b)
- NOT a = 1 – P(a)
- Constraints on what component type to return are treated literally
- Query term constraints done as follows:
  - When ranking node *x*, multiply weights along the path until we get to the constraint node
  - Then take the probability that the constraint node generated the query term

Ranking document – Query “title:bird”



P(title)P(bird|title) = 0.5*0.7 = 0.35

Interpretation of Structured Queries

- Ranking the *x* where we want *y* to contain *w* is like
  - Ranking the *document* where we want the *title* to contain *bird*
- Computing the probability that *x* generated *w* and that it was generated by *x*’s subcomponent *y*.

Summary

- Proposed a model of structured documents using language modeling
- Showed how some types of structured queries could be evaluated
- Still need to answer how to train linear interpolation weights
- Need to figure out implementation efficiency details
- Open to suggestions from you