

Distributed Information Retrieval System

贾文杰

OUTLINE

Background

Classic IR Models

Distributed IR

Opensource IR System

Our Design

Background

- Explosion of unstructured data
- Extensive applied
- Focused on retrieval of text
- Based on term frequency
- The base of many NLP tasks
- Need to deal with large scale corpus
- Many experimental IR system

Classic IR Models

- Boolean Model
- Vector Space Model
- Probabilistic Model
- Language Model

Other Models

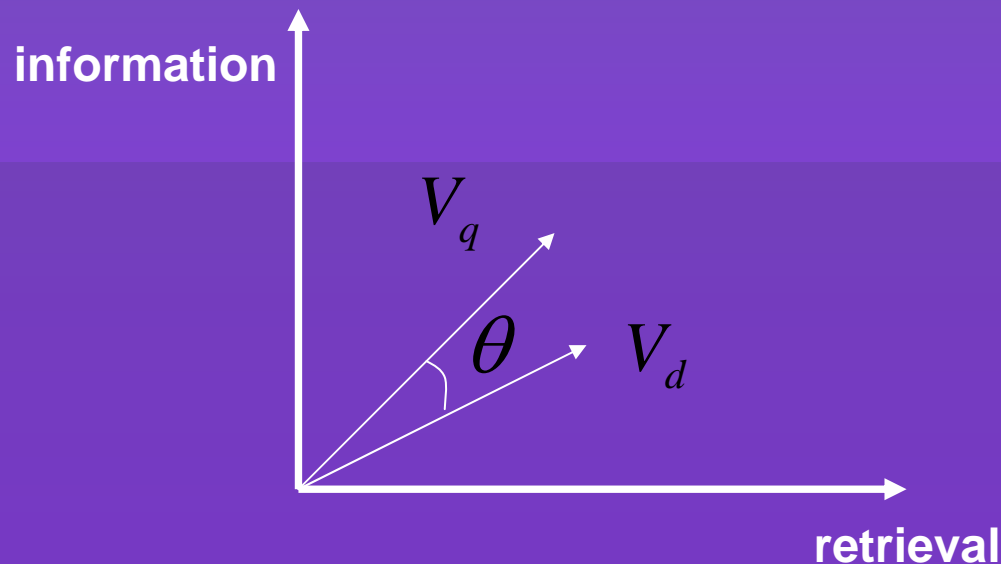
- Bayesian Network Model
- Neural Network Model

Boolean Model

- Query is formulated as a boolean combination of terms
 - A document is evaluated according to the classical rules of boolean algebra
-
- Simple structure
 - Widely applied
 - Exact match lead to bad performance
 - Either “relevant” or “non-relevant”
 - No ranking
 - No weighting

Vector Space Model

- Building Term Vectors in Document Space
- Normalization of Term Vectors
- Compute the degree of similarity
- Ranking by the degree



TFIDF Expressions

Term frequency		Document frequency		Normalization
n(natural)	tf	n(none)	df	n(no normalization)
l(logarithm)	$1 + \log(tf)$	t	$\log(N/df)$	c(cosine)
a(augmented)	$0.5 + \frac{0.5 \times tf}{\max(tf_i)}$			$\frac{1}{\sqrt{\sum W_i^2}}$

- Weighting the terms
- Find out the proximate document
- Ranking according to the similarity
- Good performance
- **Term independence assumption**

Probabilistic Model

Serious use is made of formal probability theory and statistics to arrive at the estimates of probability of relevance by which the documents are ranked

- Theoretically, ranking according to relevance
- **Divid corpus to two parts at beginning**
- **Ignore the frequency of term**
- **Term independence assumption**

Binary Independence Retrieval

Robertson and Sparck Jones

OKAPI system

The similar performance with VSM

Language Model

- Create a language model for each document
 - Rank according to the estimate of producing the query according to one model
-
- Well founded theoretical framework
 - Efficient probability estimation
 - Good effectiveness
 - Often use uni-grams
 - Smoothing is crucial

Lemur system

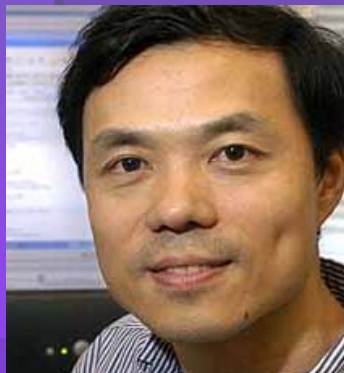
VIP of Language Model

John Lafferty

Professor

Computer Science Department

Carnegie Mellon University



Chengxiang Zhai (翟成祥)

Ph.D student of John Lafferty

Professor

University of Illinois at Urbana-Champaign



W. BRUCE CROFT

Distinguished Professor and Chair,

Department of Computer Science

and Director, Center for Intelligent Information Retrieval

University of Massachusetts, Amherst

Distributed IR

Characteristic

- Often different structured data
- Extremely large-scale corpus
- Need unified expression

Approach

- Corpus partition
- Choose the databases for retrieval
- Distribute the query to the databases
- Ranking (Merge the mid-results come from different databases)

● Corpus partition

- ◆ Duplicate the corpus to each server
- ◆ Randomly
- ◆ According to the Semanteme

● Choose the databases for retrieval

- ◆ Don't Choose
- ◆ Treat each database as a big document
- ◆ Divide each database into some segments
- ◆ Search the index of each database

● Distributed retrieval approach

- ◆ Collect global statistic information first(Centralized,Two-stage)
- ◆ Just distribute the query to each server

● Ranking

- ◆ According to the score (global)
- ◆ Circulating replacing(Merge one-by-one)
- ◆ According to the score in its database
- ◆ Multiply a database weight to document score

Opensource IR System

- Lemur (Indri) (Umass & CMU)
- Lucene (apache foundation)
- Zettair (RMIT University)
- Managing Gigabytes (The University of Melbourne)
- DataparkSearch
- SMART (Cornell University)

Search Tools Product List (89)

- ActiveSearch SiteSearch SDK
- Albert web
- Alkaline (Vestris)
- Amberfish
- ARTS PDF Search
- ASPSeek
- ASTAWARE SearchKey
- Atomica
- Atomz Search
- Autonomy Search Server
- Arexera (formerly TEC-IMS)
- BeSeen from Looksmart BooleanSearch
- BBDBot
- Blossom Hosted Search
- BRS/Search
- CGISRCH
- Compass (now iPlanet Search)
- Convera RetrievalWare
- Coveo (formerly Copernic Enterprise)
- crawl-it
- DarWin Set
- Datagold
- DeepSearch
- Dieselpoint Search
- DioWeb
- DMP Scout
- DocFather
- Doclinx TeraXML
- DolphinSearch
- dtSearch Web
- EasyAsk
- ebhath
- Educesoft ASP Search Engine
- 80-20 Discovery
- Elise Matching Engine
- Endeca Commerce, Catalog and Enterprise Search
- Engenium Semetric
- Enterprise Search (Innerprise)
- Eureka
- eVe Image Search
- Everyfind (JavaScript)
- Excalibur RetrievalWare
- Extense
- Extropia Site Search (formerly Selena Sol)
- F3DSearch

... ..

Indri **vs.** Lucene

	Indri (Lemur)	Lucene
IR Model	LM & Inference Network	VSM
Program language	C/C++	Java & many other version
Data Scale	TB (426GB)	100GB or more
Incremental indexind	Yes	Yes
Distributed IR	Yes	No
RAM requirements	User appoint	Only 1MB heap
Index speed	About 100MB/minute on Pentium4 3.0GHz	over 20MB/minute on Pentium M 1.5GHz
Index size	About 125% the size of pure text indexed	roughly 20-30% the size of text indexed

	Indri (Lemur)	Lucene
query types	Supports popular structured query operators from INQUERY	phrase queries, wildcard queries, proximity queries, range
fielded searching	Yes	Yes
Document Parser	PDF, HTML, XML, and TREC (DOC,PPT)	Many types
Stemmer	Yes	Yes
Chinese encoding	Partially support UTF-8 and GB	Yes
Chinese word segmentation	No	No
Publisher	Umass & CMU	Apache foundation
Homepage	http://www.lemurproject.org	http://lucene.apache.org/

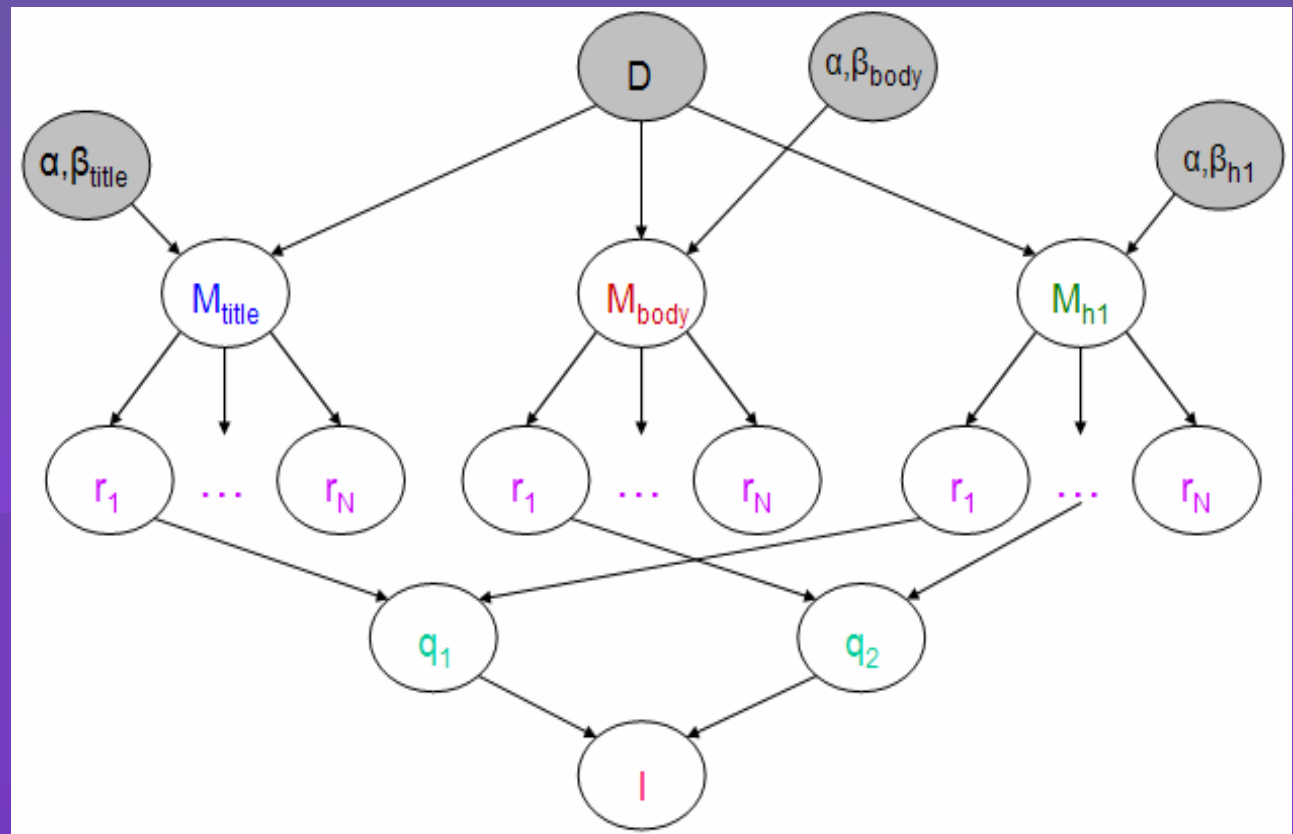
Indri Retrieval Model

Features

- Easily handles phrases (ordered and unordered)
- Can make use of multiple document representations
- Robust query language
- Explicit term weighting
- Formally well-grounded
- Highly effective
- Can be efficiently implemented

Graphical Model

- Document node
- Smoothing parameter nodes
- Model nodes
- Representation concept nodes
- Belief nodes
- Information need node

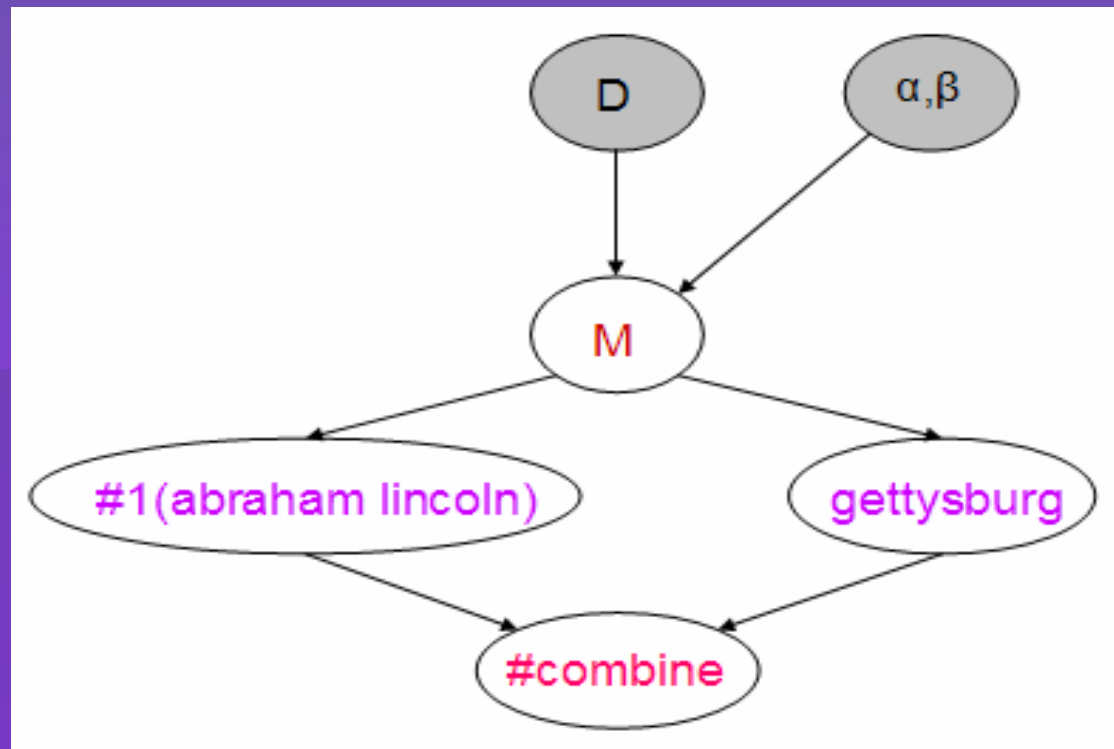


A Sample

Query: `#combine(#1(abraham lincoln) gettysburg)`

`#combine` : A
combine of terms

`#1` : exactly match



Estimation Beliefs

$$P(M | D) = \frac{P(D | M)P(M)}{\int P(D | M)P(M)dM}$$

Assume that $P(.|M)$ is distributed according to multiple-Bernoulli(theta) and $P(M)$ is distributed according to multiple-Bata(alpha,beta)

$$P(r | D) = \int P(r | M)P(M | D)dM$$

It is the expectation over the posterior $P(M|D)$

$$P(r | D) = \frac{tf_{r,D} + \alpha_r}{|D| + \alpha_r + \beta_r}$$

Use the following values:

$$\alpha_r = \mu P(r | C)$$

$$\beta_r = \mu(1 - P(r | C))$$

This ultimately leads to the following form for our term representation beliefs:

$$P(\mathbf{r} | \mathbf{D}) = \frac{\mathbf{tf}_{\mathbf{r},\mathbf{D}} + \mu P(\mathbf{r} | \mathbf{C})}{|\mathbf{D}| + \mu}$$

(Default value of mu is 2500)

Dirichlet smoothing

Combining Beliefs

The belief nodes in the network, dynamically generated based on a query, are used to combine beliefs from representation concept nodes and other belief nodes.

How the beliefs are computed

#combine

$$b_{\#combine} = \prod_{i=1}^n b_i^{\left(\frac{1}{n}\right)}$$

#weight

$$b_{\#weight} = \prod_{i=1}^n b_i^{\left(\frac{w_i}{W}\right)}$$

#or

$$b_{\#or} = 1 - \left(\prod_{i=1}^n (1 - b_i) \right)$$

#not

$$b_{\#not} = 1 - b_i$$

#max

$$b_{\#max} = \text{Max}(b_i, i = 1 \dots n)$$

#and

$$b_{\#and} = \prod_{i=1}^n b_i$$

#sum

$$b_{\#wsum} = \sum_{i=1}^n \frac{b_i}{n}$$

#wsum

$$b_{wsum} = \frac{\sum_{i=1}^n w_i b_i}{\sum_{i=1}^n w_i}$$

Things need to explore:

- Faster indexing
- Improved query processing times
- More use of document structure
- Looking into further use of complex queries
- More effective query expansion techniques for noisy data
- Document priors
- Link analysis

Our Design

- Implement a distributed IR system based on Indri system
- Add Chinese word segmentation module
- Modify some parser module to support chinese encoding(e.g. PDF,DOC,PPT)
- Implement a GUI for our system
- Implement programing sockets

That's All

Thank you!