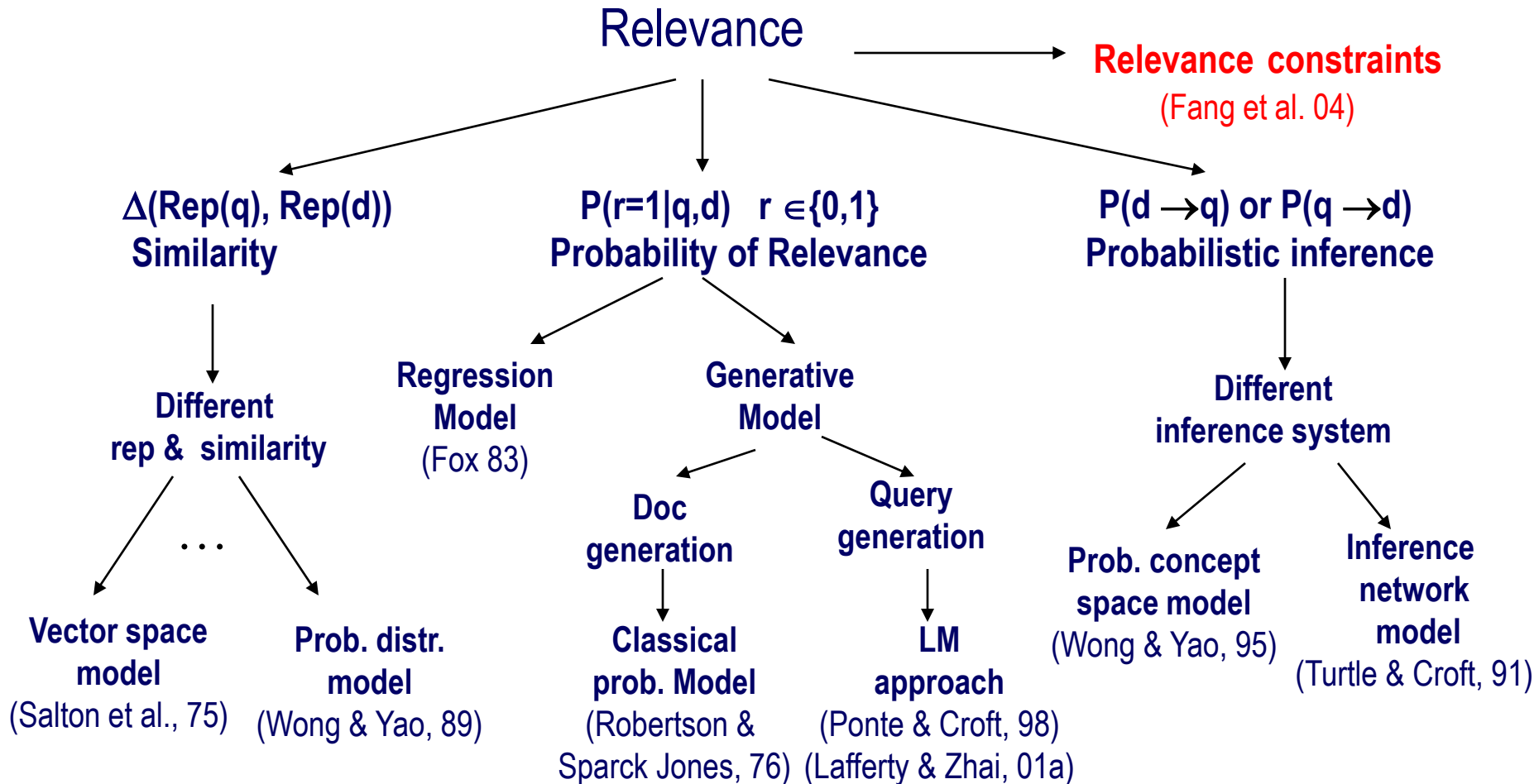
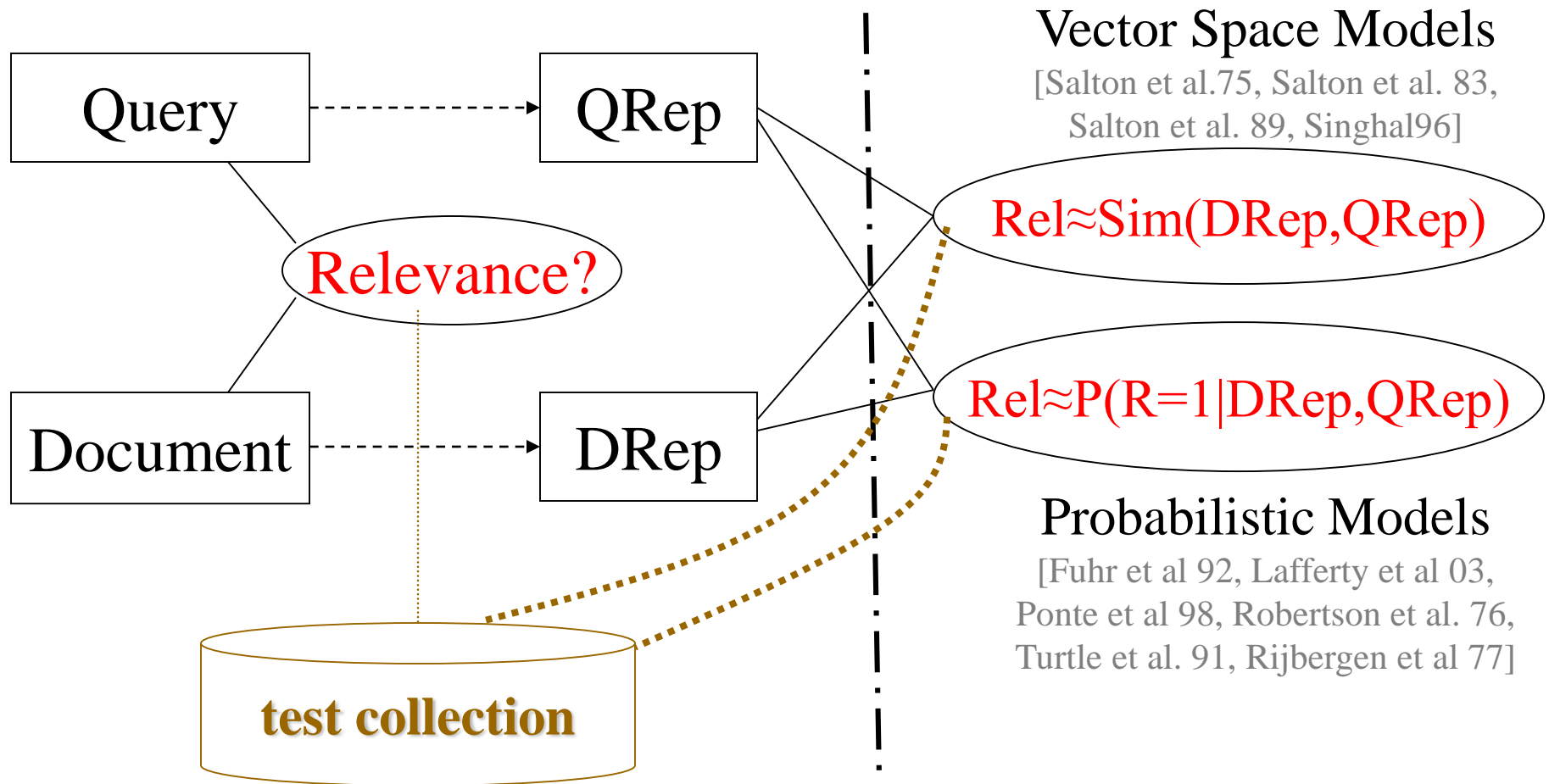


Axiomatic Retrieval Models

The Notion of Relevance



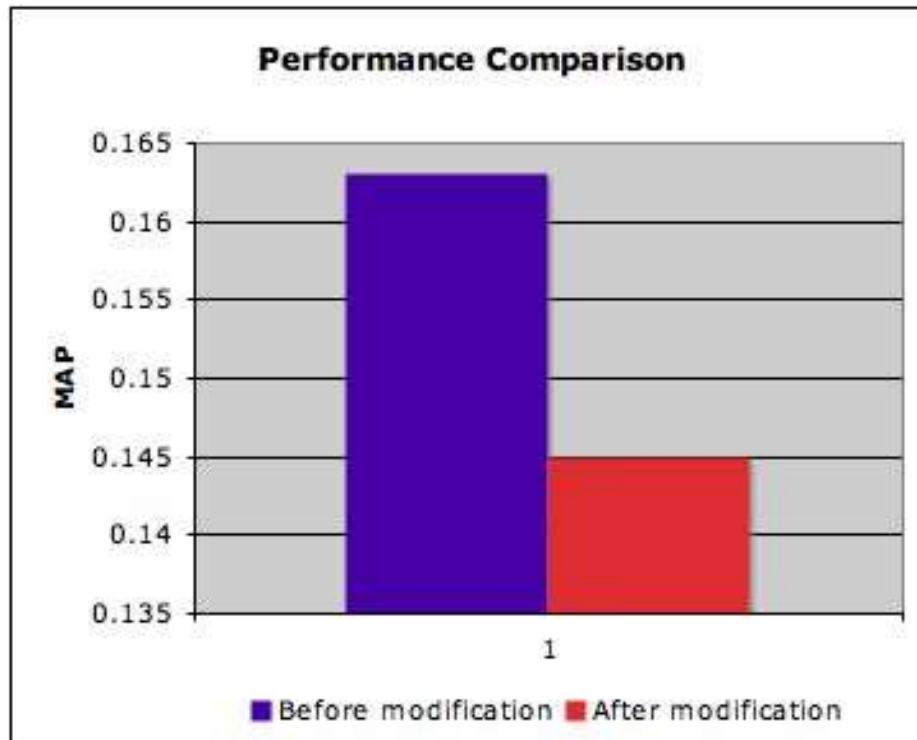
Traditional Way of Modeling Relevance



- No way to predict the performance and identify the weaknesses
- Sophisticated parameter tuning

No Way to Predict Performance

$$S(Q, D) = \sum_{t \in D \cap Q} c(t, Q) \times \log \frac{N+1}{df(t)} \times \frac{1 + \log(c(t, D))}{(1-s) + s \times \frac{|D|}{avdl}}$$



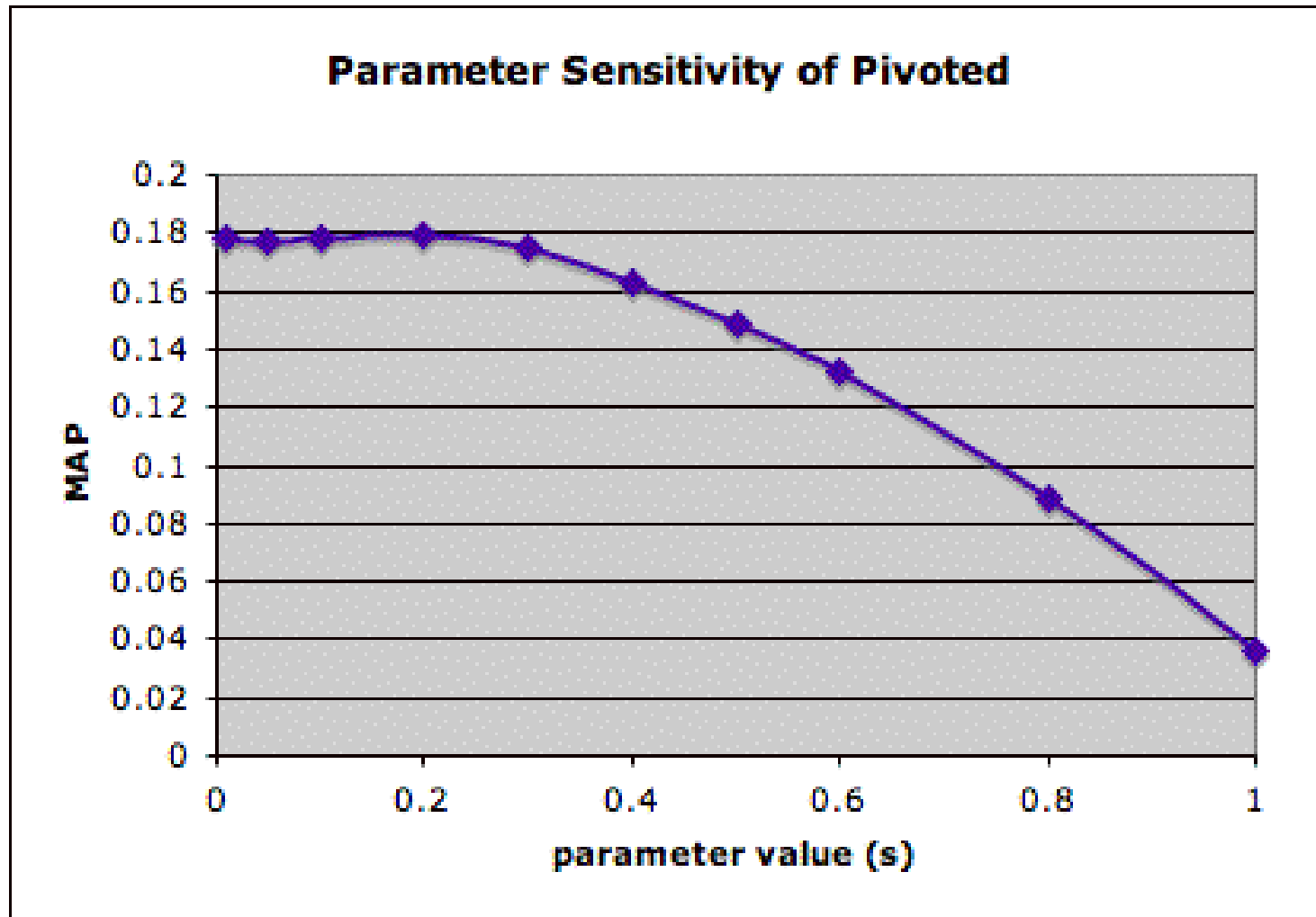
Sophisticated Parameter Tuning

$$S(Q,D) = \sum_{t \in Q \cap D} \log \frac{N - df(t) + 0.5}{df(t)} \cdot \frac{(k_1 + 1) \cdot c(t,D)}{c(t,D) + k_1((1 - b) + b \cdot \frac{|D|}{avdl})} \cdot \frac{(k_3 + 1) \cdot c(t,Q)}{k_3 + c(t,Q)}$$

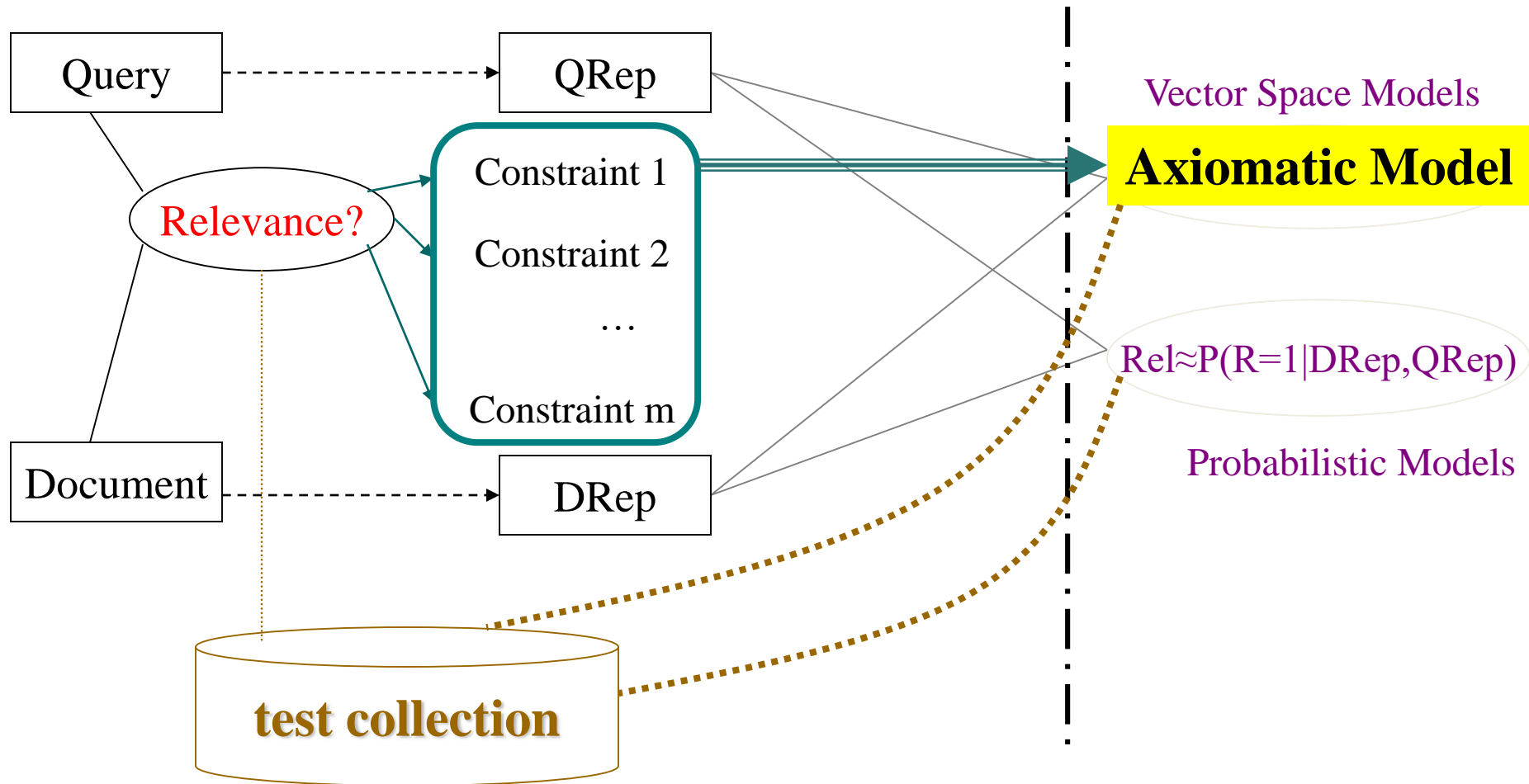
“**k₁**, **b** and **k₃** are parameters which depend on the nature of the queries and possibly on the database;
k₁ and **b** default to 1.2 and 0.75 respectively,
but smaller values of **b** are sometimes advantageous;
in long queries **k₃** is often set to 7 or 1000.”

[Robertson et al. 1999]

High Parameter Sensitivity

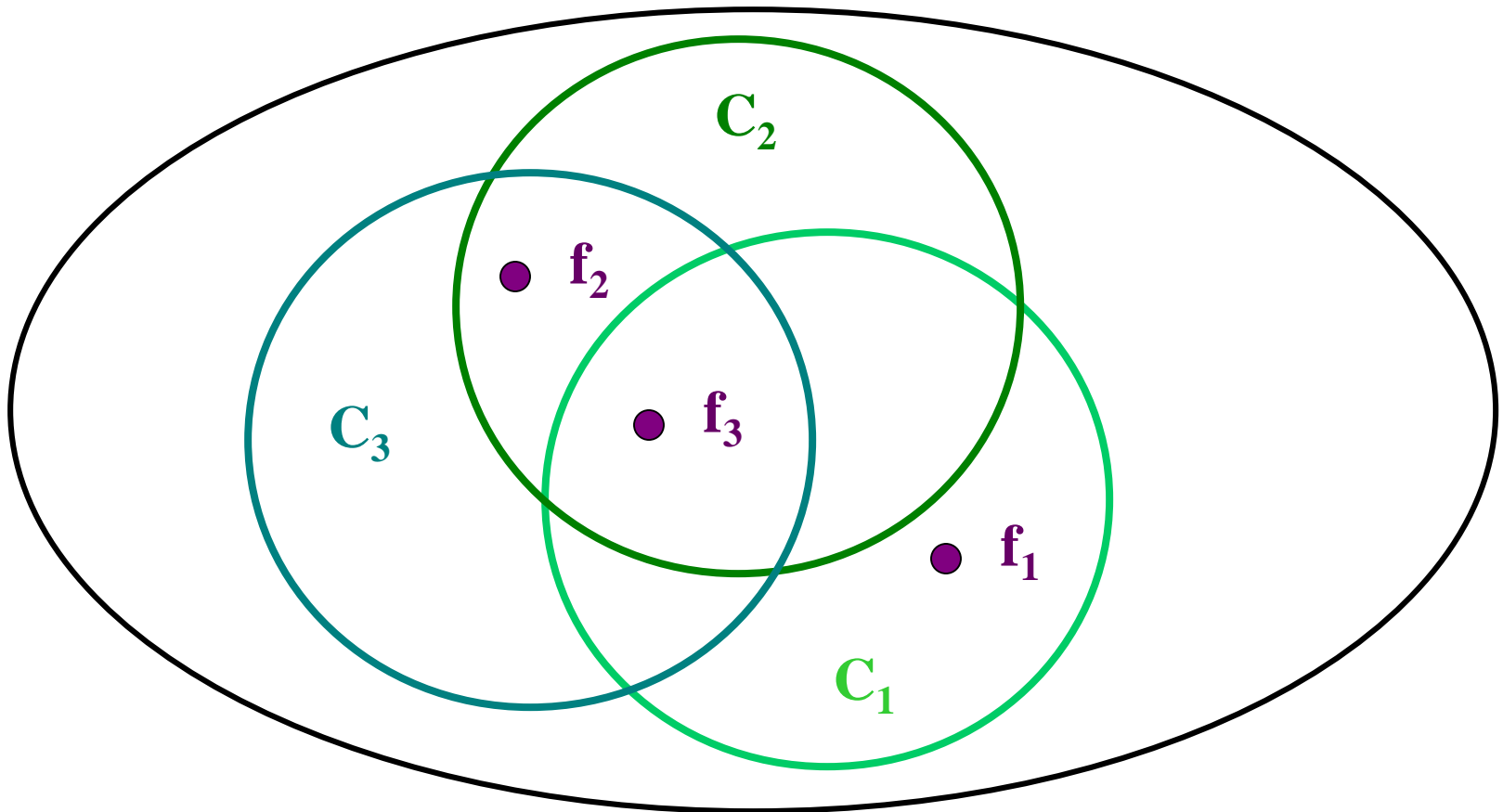


Axiomatic Approach to Relevance Modeling



Basic Idea of Axiomatic Approach

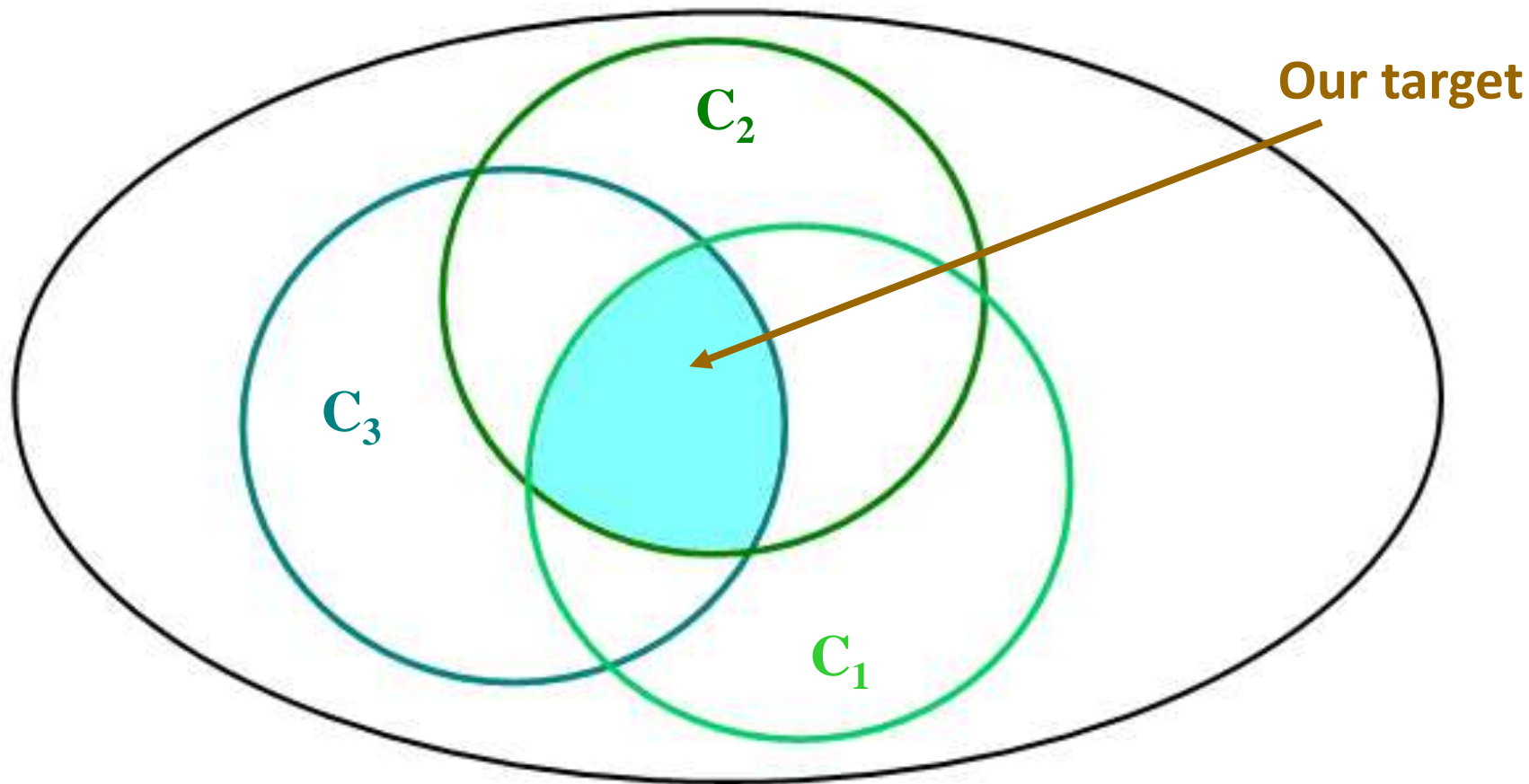
A set of **retrieval constraints** that any reasonable retrieval function should satisfy



The **search space** for all possible retrieval functions

Basic Idea of Axiomatic Approach

A set of **retrieval constraints** that any reasonable retrieval function should satisfy



The **search space** for all possible retrieval functions

An Axiomatic Framework for Retrieval Functions

- Component 1: Constraints to be satisfied by an effective retrieval function
- Component 2: Function space that allows us to efficiently search for an effective function

Implementation of Component 1:

Question:

**How do we define retrieval
constraints?**

The Three Major Heuristics

- Pivoted Normalization Method

$$\sum_{w \in q \cap d} \frac{1 + \ln(1 + \ln(c(w, d)))}{(1-s) + s \frac{|d|}{avdl}} \cdot c(w, q) \cdot \ln \frac{N+1}{df(w)}$$

- Dirichlet Prior Method

Inverse Document Frequency

$$\sum_{w \in q \cap d} c(w, q) \times \ln\left(1 + \frac{c(w, d)}{\mu \cdot p(w|C)}\right) + |q| \ln \frac{\mu}{\mu + |d|}$$

Term Frequency

Document Length Normalization

- Okapi Method

$$\sum_{w \in q \cap d} \ln \frac{N - df(w) + 0.5}{df(w) + 0.5} \cdot \frac{(k_1 + 1) \times c(w, d)}{k_1((1-b) + b \frac{|d|}{avdl}) + c(w, d)} \cdot \frac{(k_3 + 1) \times c(w, q)}{k_3 + c(w, q)}$$

Term Frequency Constraints (TFC1)

TF weighting heuristic I:

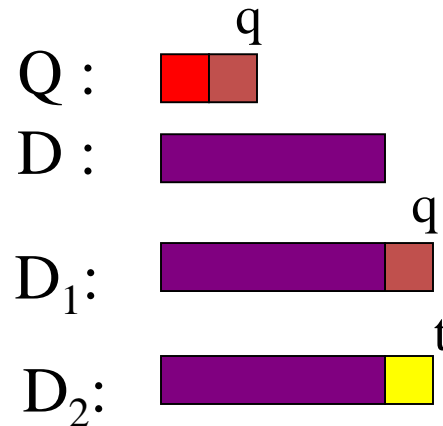
Give a higher score to a document with more occurrences of a query term.

- *TFC1*

Let Q be a query and D be a document.

If $q \in Q$ and $t \notin Q$,

then $S(Q, D \cup \{q\}) > S(Q, D \cup \{t\})$



$$S(Q, D_1) > S(Q, D_2)$$

Term Frequency Constraints (TFC2)

TF weighting heuristic II:

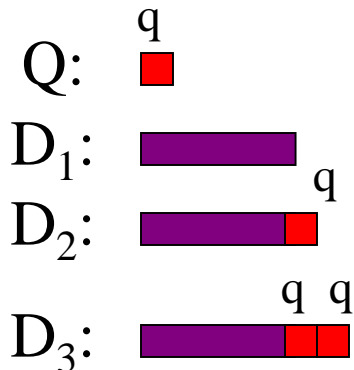
Require that the amount of increase in the score due to adding a query term must decrease as we add more terms.

- *TFC2*

Let Q be a query with only one query term q .

Let D_1 be a document.

then $S(Q, D_1 \cup \{q\}) - S(Q, D_1) > S(Q, D_1 \cup \{q\} \cup \{q\}) - S(Q, D_1 \cup \{q\})$



$$S(Q, D_2) - S(Q, D_1) > S(Q, D_3) - S(Q, D_2)$$

Term Frequency Constraints (TFC3)

TF weighting heuristic III:

if two documents have the same total occurrences of all query terms
and all the query terms have same term discrimination value, a higher
score will be given to the document covering more distinct query terms

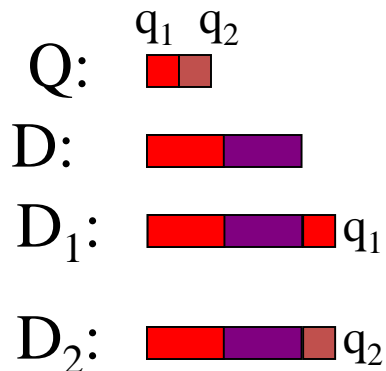
- *TFC3*

Let Q be a query with two query terms $Q = \{q_1, q_2\}, td(q_1) = td(q_2)$

Let D be a document.

If $q_1 \in D$ and $q_2 \notin D$

then $S(Q, D \cup \{q_1\}) < S(Q, D \cup \{q_2\})$



$$S(Q, D_1) < S(Q, D_2)$$

Term Discrimination Constraint (TDC)

Term discrimination heuristic:

penalize the terms popular in the collection

- *TDC*

Let Q be a query with two query terms $Q = \{q_1, q_2\}$

Let D be a document.

Let D_1 be a document, containing only q_1
Let D_2 be a document, containing only q_2 } $|D_1| = |D_2|$

If $td(q_1) > td(q_2)$

then $S(Q, D \cup D_1) > S(Q, D \cup D_2)$

Length Normalization Constraints (LNCs)

Document length normalization heuristics:

Penalize long documents(LNC1);

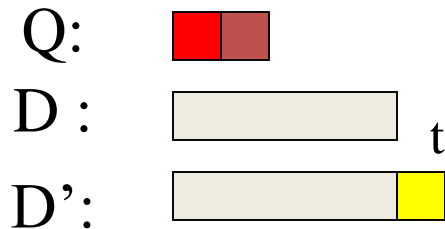
Avoid over-penalizing long documents (LNC2) .

- **LNC1**

Let Q be a query and D be a document.

If t is a non-query term,

then $S(Q, D \cup \{t\}) \leq S(Q, D)$



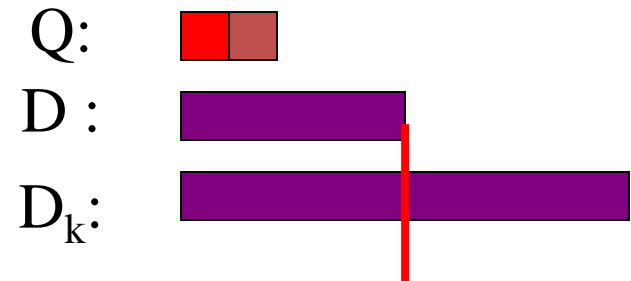
$$S(Q, D') \leq S(Q, D)$$

- **LNC2**

Let Q be a query and D be a document.

If $D \cap Q \neq \emptyset$, and D_k is constructed by concatenating D with itself k times,

then $S(Q, D_k) \geq S(Q, D)$



$$S(Q, D_k) \geq S(Q, D)$$

TF-LENGTH Constraint (TF-LNC)

TF-LN heuristic:

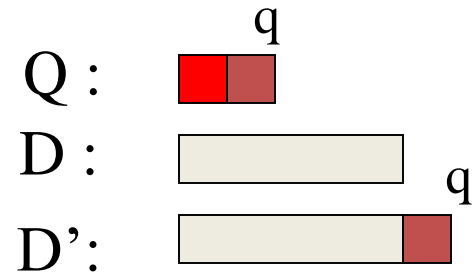
Regularize the interaction of TF and document length.

- *TF-LNC*

Let Q be a query and D be a document.

If q is a query term,

then $S(Q, D \cup \{q\}) \geq S(Q, D)$



$$S(Q, D') \geq S(Q, D)$$

Summary of intuitions for each formalized constraint

Constraints	Intuitions
TFC1	to favor a document with more occurrence of a query term
TFC2	to ensure that the amount of increased score due to adding a query term must decrease as more terms are added
TFC3	to favor a document matching more distinct query terms
TDC	to penalize the words popular in the collection and assign higher weights to discriminative terms
LNC1	to penalize a long document
LNC2, TF-LNC	to avoid over-penalizing a long document
TF-LNC	to regulate the interaction of TF and document length

Analytical Evaluation

Retrieval Formula	<i>TFCs</i>	<i>TDC</i>	<i>LNC1</i>	<i>LNC2</i>	<i>TF-LNC</i>
Pivoted Norm.	Yes	Yes	Yes	Conditional*	Conditional*
Dirichlet Prior	Yes	Yes	Yes	Conditional	Yes
Okapi (original)	Conditional	Yes	Conditional	Conditional	Conditional
Okapi (modified)	Yes	Yes	Yes	Yes	Yes

Analytical Evaluation

Pivoted Normalization & TFC1

$$S(Q, D) = \sum_{t \in Q \cap D} \frac{1 + \ln(1 + \ln(c(t, D)))}{1 - s + s \frac{|D|}{avdl}} \cdot c(t, Q) \cdot \ln \frac{N + 1}{df(t)}$$

- TFC1:

Let Q be a query and D be a document.

If $q \in Q$ and $w \notin Q$, then $S(Q, D \cup \{q\}) > S(Q, D \cup \{w\})$

Let $D_1 = D \cup \{q\}$, $D_2 = D \cup \{w\}$

$S(Q, D_1) > S(Q, D_2)$ since D_1 has an extra matched term

Analytical Evaluation

Pivoted Normalization & LNC1

$$S(Q, D) = \sum_{t \in Q \cap D} \frac{1 + \ln(1 + \ln(c(t, D)))}{1 - s + s \frac{|D|}{avdl}} \cdot c(t, Q) \cdot \ln \frac{N + 1}{df(t)}$$

- LNC1:

Let Q be a query and D be a document.

If w is a non-query term, then $S(Q, D \cup \{w\}) \leq S(Q, D)$

Let $D_1 = D \cup \{w\}$

$$\begin{aligned} S(Q, D_1) &= \sum_{t \in Q \cap D_1} \frac{1 + \ln(1 + \ln(c(t, D_1)))}{1 - s + s \frac{|D_1|}{avdl}} \cdot c(t, Q) \cdot \ln \frac{N + 1}{df(t)} \\ &= \sum_{t \in Q \cap D} \frac{1 + \ln(1 + \ln(c(t, D)))}{1 - s + s \frac{|D| + 1}{avdl}} \cdot c(t, Q) \cdot \ln \frac{N + 1}{df(t)} \end{aligned}$$

Thus $S(Q, D_1) < S(Q, D)$

Analytical Evaluation

Okapi & TFC1

$$S(Q, D) = \sum_{t \in Q \cap D} \left(\ln \frac{N - df(t) + 0.5}{df(t) + 0.5} \times \frac{(k_1 + 1)c(t, D)}{k_1 \left(1 - b + b \frac{|D|}{avdl} \right) + c(t, D)} \times \frac{(k_3 + 1) \cdot c(t, Q)}{k_3 + c(t, Q)} \right)$$

- TFC1:

Let Q be a query and D be a document.

If $q \in Q$ and $w \notin Q$, then $S(Q, D \cup \{q\}) > S(Q, D \cup \{w\})$

If $df(t) > \frac{N}{2}$, TFC1 is not satisfied

Analytical Evaluation

Okapi & LNC1

$$S(Q, D) = \sum_{t \in Q \cap D} \left(\ln \frac{N - df(t) + 0.5}{df(t) + 0.5} \times \frac{(k_1 + 1)c(t, D)}{k_1 \left(1 - b + b \frac{|D|}{avdl} \right) + c(t, D)} \times \frac{(k_3 + 1) \cdot c(t, Q)}{k_3 + c(t, Q)} \right)$$

- LNC1:

Let Q be a query and D be a document.

If w is a non-query term, then $S(Q, D \cup \{w\}) \leq S(Q, D)$

If $df(t) > \frac{N}{2}$, LNC1 is not satisfied

**Is constraint analysis related to the
performance of a retrieval
function?**

Benefits of Constraint Analysis

- Provide an approximate bound for the parameters
 - A constraint may be satisfied only if the parameter is within a particular interval.
- Compare different formulas analytically without experimentations
 - When a formula does not satisfy the constraint, it often indicates non-optimality of the formula.
- Suggest how to improve the current retrieval models
 - Violation of constraints may pinpoint where a formula needs to be improved.

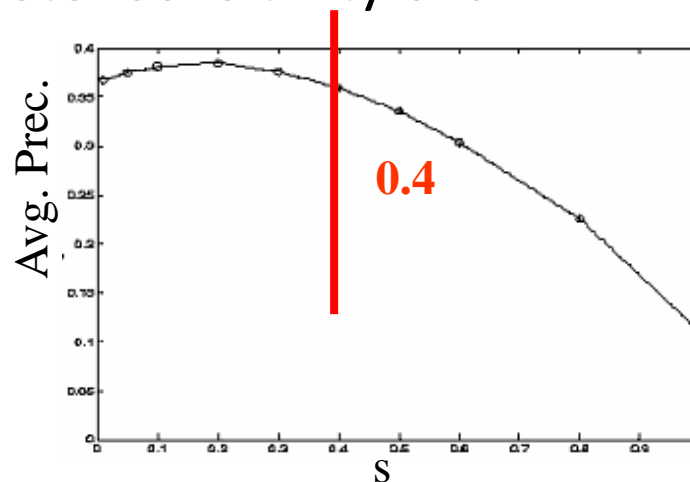
Conditional Satisfaction of Constraints → Parameter Bounds

- Pivoted Normalization Method LNC2 → $s < 0.4$

Optimal s (for average precision)

	AP	DOE	FR	ADF	Web	Trec 7	Trec 8
LK	0.2	0.2	0.05	0.2	---	---	---
SK	0.01	0.2	0.01	0.05	0.01	0.05	0.05
LV	0.3	0.3	0.1	0.2	0.2	0.2	0.2
SV	0.2	0.3	0.1	0.2	0.1	0.1	0.2

Parameter sensitivity of s



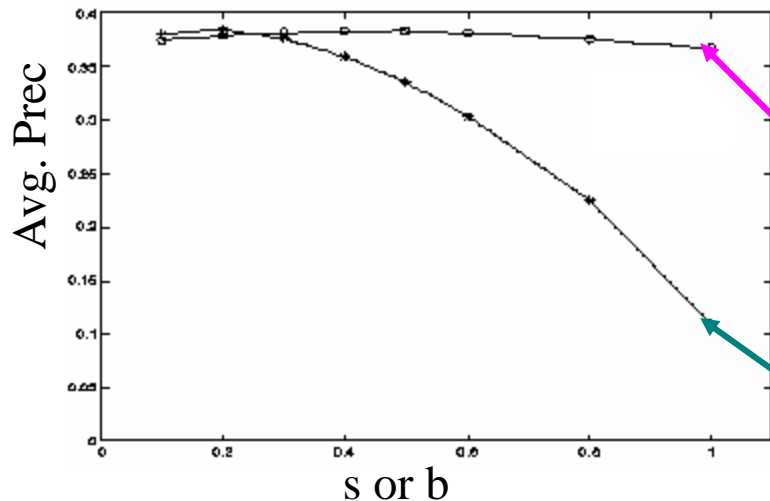
Violation of Constraints → Poor Performance

- Okapi Method

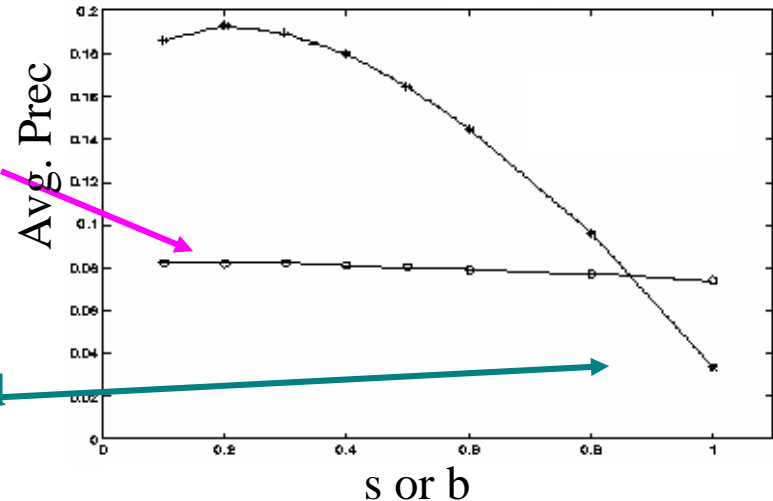
$$\sum_{w \in q \cap d} \ln \frac{N - df(w) + 0.5}{df(w) + 0.5} \cdot \frac{(k_1 + 1) \times c(w, d)}{k_1((1 - b) + b \frac{|d|}{avdl}) + c(w, d)} \cdot \frac{(k_3 + 1) \times c(w, q)}{k_3 + c(w, q)}$$

Negative when $df(w)$ is large → Violate many constraints

keyword query



verbose query



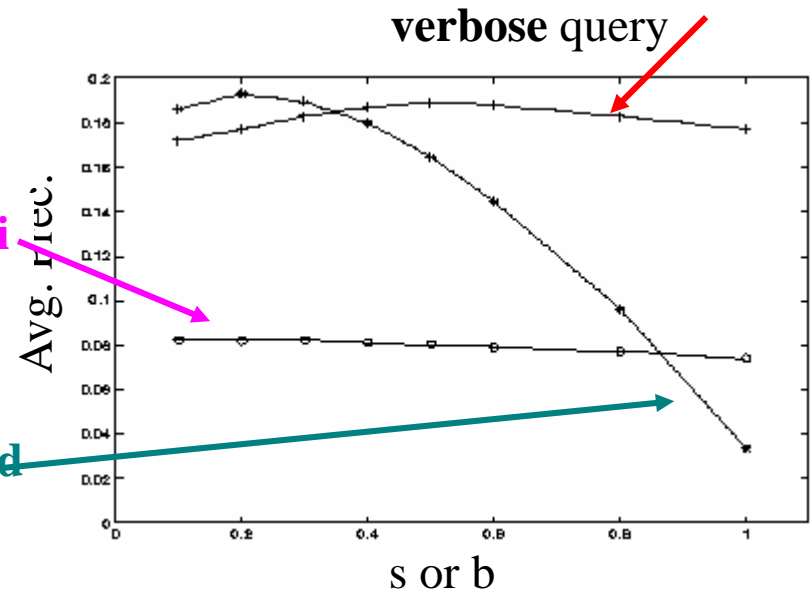
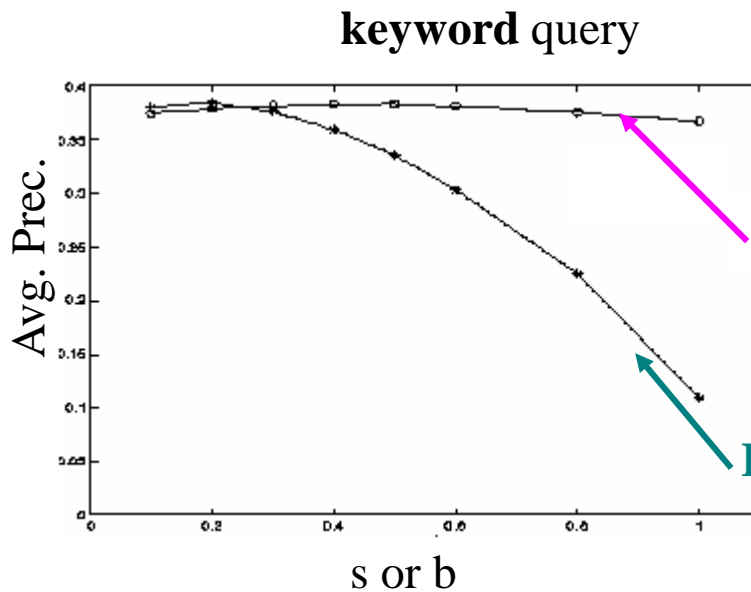
Constraints Analysis → Guidance for Improving a Retrieval Function

- Modified Okapi Method

$$\sum_{w \in q \cap d} \ln \frac{N - df(w) + 0.5}{df(w) + 0.5} \cdot \frac{\ln \frac{N+1}{df}}{k_1((1-b) + b \frac{|d|}{avdl}) + c(w, d)} \cdot \frac{(k_1 + 1) \times c(w, d)}{k_3 + c(w, q)}$$

Make Okapi satisfy more constraints; expected to help verbose queries

Modified Okapi



Implementation of Component 2:

Question:

**How can we define a function space
that can be searched efficiently?**

Component 2: Function Space

$$Q = \{q_1, q_2, \dots, q_m\}$$

Bag of terms

$$D = \{d_1, d_2, \dots, d_n\}$$



$$S: Q \times D \rightarrow \mathfrak{R}$$



Function Space

Define the function space *inductively*



Inductive Definition of Function Space

Primitive weighting function



Q:  q
 $S(Q, D) = f(q, d) = \text{weight}(q)$
 D:  d (=q)

Q:  q
 $S(Q, D) = f(q, d)$
 D:  d ($\neq q$) = $\text{penalty}(q, d)$

Query growth function

Q' :  q' $S(Q', D) = S(Q \cup \{q'\}, D) = h(S(Q, D), S(\{q'\}, D), q', D, Q)$
 D: 

Document growth function

Q:  $S(Q, D') = S(Q, D \cup \{d'\}) = g(S(Q, D), S(Q, \{d'\}), d', D, Q)$
 D':  d'

Need to ensure that this indeed defines a function

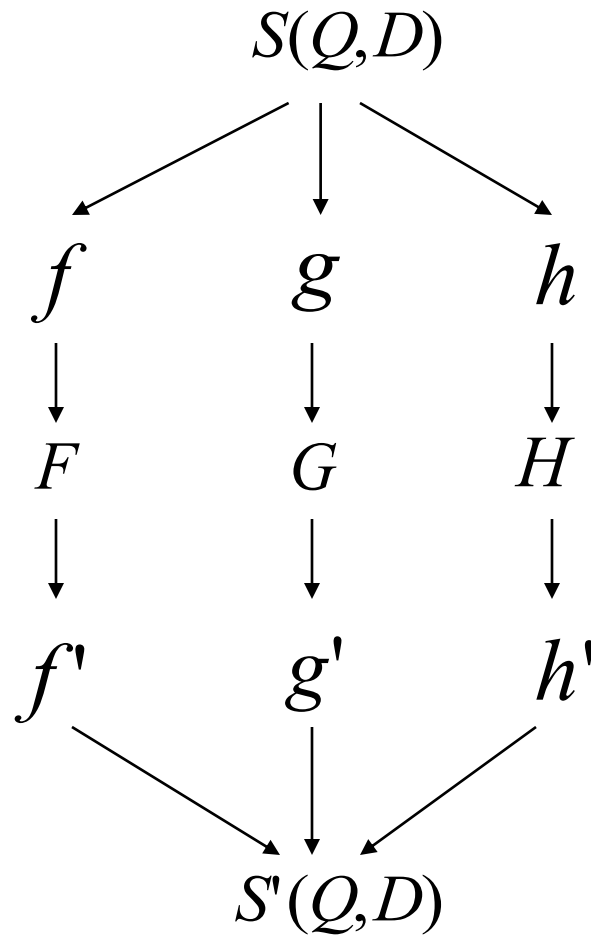
Derivation of New Retrieval Functions

decompose

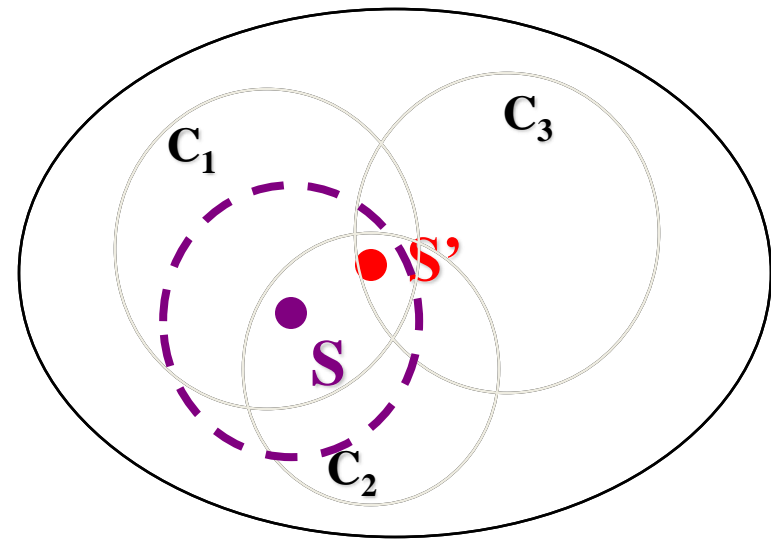
generalize

constrain

assemble



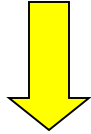
existing function



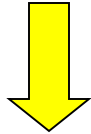
new function

Derivation of New Retrieval Functions

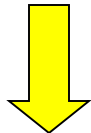
Decompose the existing retrieval functions



Generalize each component function



**Use constraints to find some alternative implementation
of a component function**

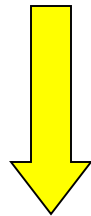


**Assemble the new component functions to form a new
retrieval function.**

Component Function 1:

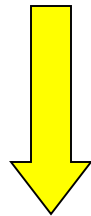
Primitive Weighting Function

Decompose the existing retrieval functions



Roughly IDF

Generalize the component function



Choice 1: $weight(q) = P(rel \mid occ)$

Choice 2: $weight(q) = \log \frac{P(occ \cap rel)}{P(occ)P(rel)}$

Use constraints to find some alternative implementation

Choice 1: $weight(q) = \left(\frac{N}{df(q)}\right)^k$ EXP

Choice 2: $weight(q) = \log \frac{N}{df(q)}$ LOG

Primitive Weighting Function of Pivoted Normalization

$$S(Q, D) = \sum_{t \in Q \cap D} \frac{1 + \ln(1 + \ln(c(t, D)))}{1 - s + s \frac{|D|}{avdl}} \cdot c(t, Q) \cdot \ln \frac{N + 1}{df(t)}$$

$$S(\{q\}, \{q\}) = \frac{1 + \ln(1 + \ln(1))}{1 - s + s \cdot \frac{1}{avdl}} \cdot 1 \cdot \ln \frac{N + 1}{df(q)} = \frac{1}{1 - s + s \cdot \frac{1}{avdl}} \ln \frac{N + 1}{df(q)}$$

$$S(\{q\}, \{d\}) = 0$$

Primitive Weighting Function of Okapi

$$S(Q, D) = \sum_{t \in Q \cap D} \left(\ln \frac{N - df(t) + 0.5}{df(t) + 0.5} \times \frac{(k_1 + 1)c(t, D)}{k_1 \left(1 - b + b \frac{|D|}{avdl} \right) + c(t, D)} \times \frac{(k_3 + 1).c(t, Q)}{k_3 + c(t, Q)} \right)$$

$$S(\{q\}, \{q\}) = \ln \frac{N - df(q) + 0.5}{df(q) + 0.5} \times \frac{(k_1 + 1).1}{k_1 \left(1 - b + b \frac{1}{avdl} \right) + 1} \times \frac{(k_3 + 1).1}{k_3 + 1}$$

$$= \ln \frac{N - df(q) + 0.5}{df(q) + 0.5} \times \frac{k_1 + 1}{k_1 \left(1 - b + \frac{b}{avdl} \right) + 1}$$

$$S(\{q\}, \{d\}) = 0$$

Component Function 2: Query Growth Function

Follow the existing retrieval functions:

$$S(Q \cup \{q\}, D) = S(Q, D) + \alpha S(\{q\}, D)$$

$\alpha = 1$ (*same as Pivoted & Dirichlet*)

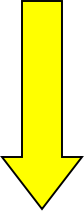
Query Growth Function of Pivoted Normalization

$$S(Q, D) = \sum_{t \in Q \cap D} \frac{1 + \ln(1 + \ln(c(t, D)))}{1 - s + s \frac{|D|}{avdl}} \cdot c(t, Q) \cdot \ln \frac{N + 1}{df(t)}$$

$$S(Q \cup \{q\}, D) = S(Q, D) + S(\{q\}, D)$$

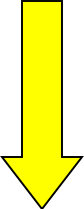
Component Function 3: Document Growth Function

Decompose the existing functions Pivoted Normalization



$$\frac{1 - s + s \frac{|D|}{avdl}}{1 - s + s \frac{|D| + 1}{avdl}} S(Q, D) + \frac{1 - s + s \frac{1}{avdl}}{1 - s + s \frac{|D| + 1}{avdl}} \left[\ln(1 + \ln(c(d, D) + 1)) - \ln(1 + \ln(c(d, D))) \right] \cdot S(Q, \{d\})$$

Generalize the component function of existing functions



$$\lambda_1(|D|) S(Q, D) + \lambda_2(|D|) \cdot \alpha(c(d, D)) \cdot S(Q, \{d\})$$

Use constraints to find some alternative implementation

$$\lambda_1(k) = \frac{k + avdl / s}{k + 1 + avdl / s}, \lambda_2(k) = \frac{1 + avdl / s}{k + 1 + avdl / s}$$

Some Derived Formulas

		Primitive Weighting Function	
		LOG	EXP
Doc. Growth Func.	PN-variation	$\sum_{t \in Q \cap D} c(t, Q) \cdot TF(c(t, D)) \cdot LN(D) \cdot LW(t)$	$\sum_{t \in Q \cap D} c(t, Q) \cdot TF(c(t, D)) \cdot LN(D) \cdot EW(t)$
	Okapi-variation	$\sum_{t \in Q \cap D} c(t, Q) \cdot TF_LN(c(t, D), D) \cdot LW(t)$	$\sum_{t \in Q \cap D} c(t, Q) \cdot TF_LN(c(t, D), D) \cdot EW(t)$
	DP-variation	$\sum_{t \in Q \cap D} c(t, Q) \cdot TF(c(t, D)) \cdot LW(t) - \gamma(D , Q)$	$\sum_{t \in Q \cap D} c(t, Q) \cdot TF(c(t, D)) \cdot EW(t) - \gamma(D , Q)$

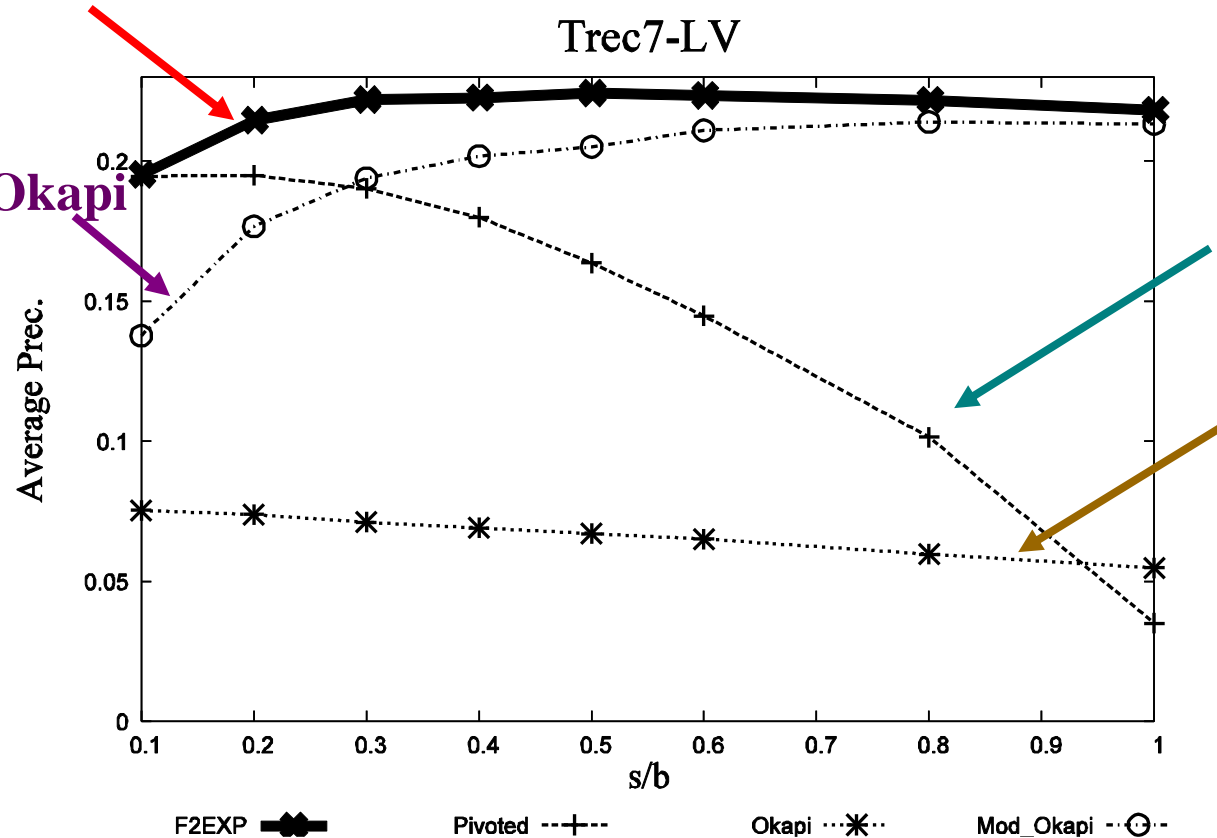
$$S(Q, D) = \sum_{t \in Q \cap D} c(t, Q) \cdot \left(\frac{N}{df(t)} \right)^k \cdot \frac{c(t, D)}{c(t, D) + s + \frac{s \cdot |D|}{avdl}}$$

Re-parameterization of Okapi TF with one parameter

Derived Axiomatic Retrieval Function is More Robust

Derived Formula

Modified Okapi



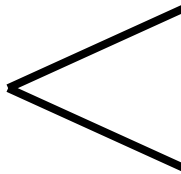
Pivoted

Okapi

Adding Semantic Term Matching

dog

A **book** is a collection of papers with text, pictures, usually bound together along one edge within covers. A **book** is also a literary work or a main division of such a work. A **book** produced in electronic format is known as an e-book.



Training **puppies** is not always easy: it requires work. **Puppies** should be touched and held from birth, although only briefly and occasionally until their eyes and ears open. Otherwise the **puppy** may become vicious.

General Approach to Semantic Term Matching

- Select semantic similar terms
- Expand original query with the selected terms

dog	1
puppy	0.5
doggy	0.5
hound	0.5
bone	0.1

Key challenge:

How to weight selected terms?

The proposed axiomatic approach provides guidance on how to weight terms appropriately.

Semantic Term Matching Constraints (STMC1)

- Semantic similarity function: $s(t, u) \in [0, +\infty]$
- t is semantically more similar to u than to v iff $s(t, u) > s(t, v)$
- A term has the highest similarity to itself: $\forall u \neq t, s(t, t) > s(t, u)$

Semantic term matching heuristic I:

Give a higher score to a document with a term that is more semantically related to a query term

- STMC1

Let $Q = \{q\}$ be a query with only one term q .

Let $D1 = \{d1\}$ and $D2 = \{d2\}$, where $q \neq d1$ and $q \neq d2$.

If $s(q, d1) > s(q, d2)$, then $S(Q, D1) > S(Q, D2)$.

Semantic Term Matching Constraints (STMC2)

Semantic term matching heuristic II:

Matching an original query term exactly should always contribute no less to the relevance score than matching a semantically related term

- STMC2

Let $Q = \{q\}$ be a single term query

Let d be a non-query term such that $s(q, d) > 0$

If $D1$ and $D2$ are two documents where:

$|D1| = 1, c(q, D1) = 1, |D2| = k$ and $c(d, D2) = k$ ($k \geq 1$)

then $S(Q, D1) \geq S(Q, D2)$.

Semantic Term Matching Constraints (STMC3)

Semantic term matching heuristic III:

Consider a query with two equally important terms q_1 and q_2 . Suppose a document D_1 matches q_1 n (> 1) times, but does not match q_2 or any of its semantically related terms. If we change one of the occurrences of q_1 in D_1 to a term semantically related to q_2 to form a document D_2 , D_2 should not have a lower score than D_1 , because D_2 covers more distinct query terms than D_1 .

- STMC3

Let $Q = \{q_1, q_2\}$ be a query

Let d be a non-query term such that $s(q_2, d) > 0$

Let D_1 and D_2 be two documents.

If $|D_1| = |D_2| > 1$, $S(\{q_1\}, \{q_1\}) = S(\{q_2\}, \{q_2\})$,

$c(q_1, D_1) = |D_1|$, $c(q_1, D_2) = |D_2| - 1$ and $c(d, D_2) = 1$,

then $S(Q, D_1) \leq S(Q, D_2)$.

Extension Based on STMCs

- Generalized primitive weighting function:

$$S(\{q\}, \{d\}) = w(q) \times f(s(q, d))$$

f is a monotonically increasing function
 $\forall q \in Q, f(s(q, q)) = 1$

e.g.

$$f(s(q, d)) = \frac{s(q, d)}{s(q, q)} \times \lambda(q, d)$$

$$\lambda(q, d) = \begin{cases} 1 & q = d \\ \beta & q \neq d \end{cases}$$

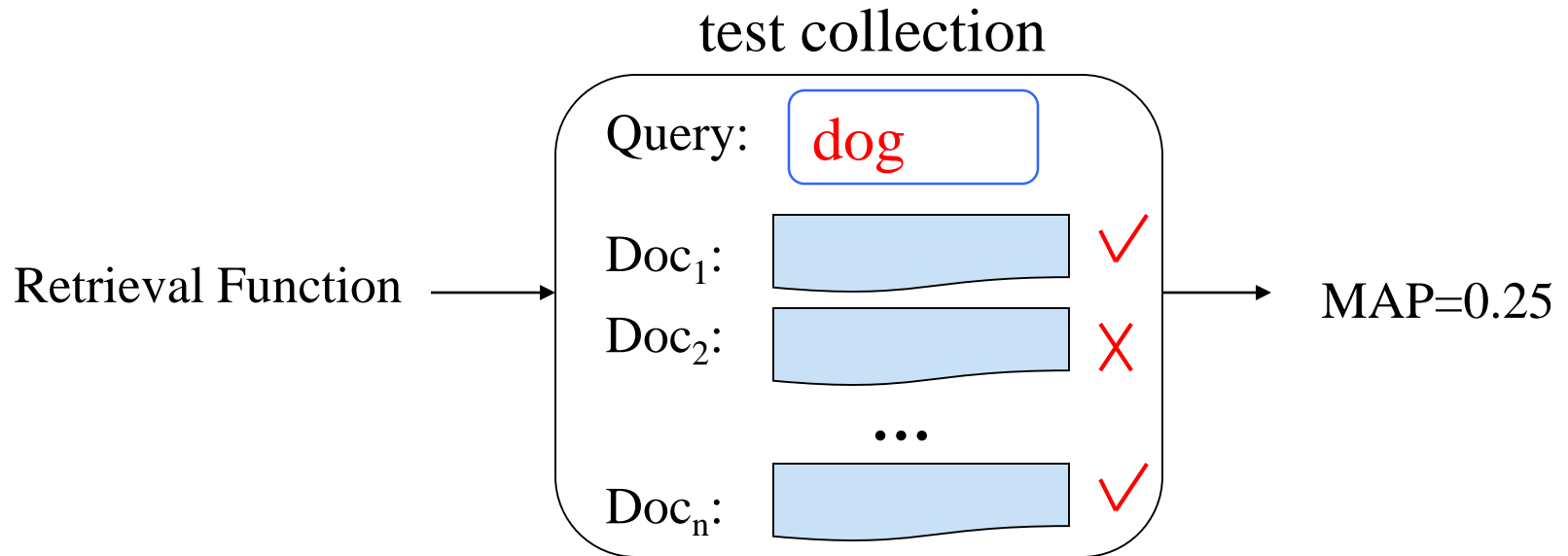
regulate the weighting

Effectiveness of Semantic Term Matching

	ROBUST04		ROBUST05	
	MAP	P@20	MAP	P@20
Syntactic term matching (baseline)	0.248	0.352	0.192	0.379
Semantic term matching	0.302 (21.8%)	0.399	0.292 (51.0%)	0.502

Diagnostic evaluation for IR models

Existing evaluation provides little explanation for the performance differences



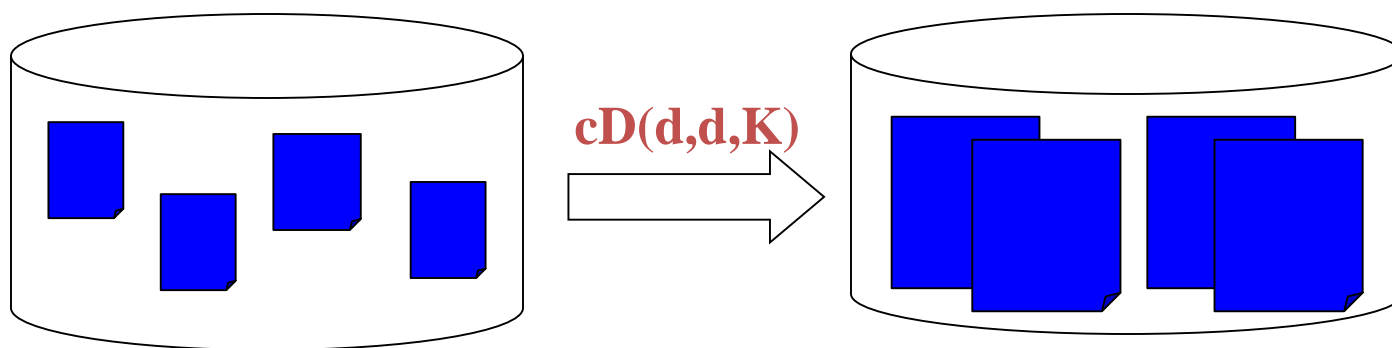
	trec8	wt-2g	fr88-89
Pivoted	0.244	0.288	0.218
Dirichlet	0.257	0.302	0.202

How to diagnose weaknesses and strengths of retrieval functions?

Relevance-Preserving Perturbations

Perturb term statistics in documents and keep relevance status

document scaling perturbation:



concatenate every document with itself K times

Some Notations

- D : Document set
- D_r : Relevant document set
- D_n : Non-relevant document set
- V_n : Noise term vocabulary

Basic Perturbation Operators

Name	Semantic	Operator
Term addition	Add K occurrences of term t to document D	$aT(t, D, K)$
Term deletion	Delete K occurrences of term t from document D	$dT(t, D, K)$
Document addition	Add document D to collection K times	$aD(D, K)$
Document deletion	Delete document D from the collection	$dD(D)$
Document concatenation	Concatenate document D_1 with D_2 K times	$cD(D_1, D_2, K)$

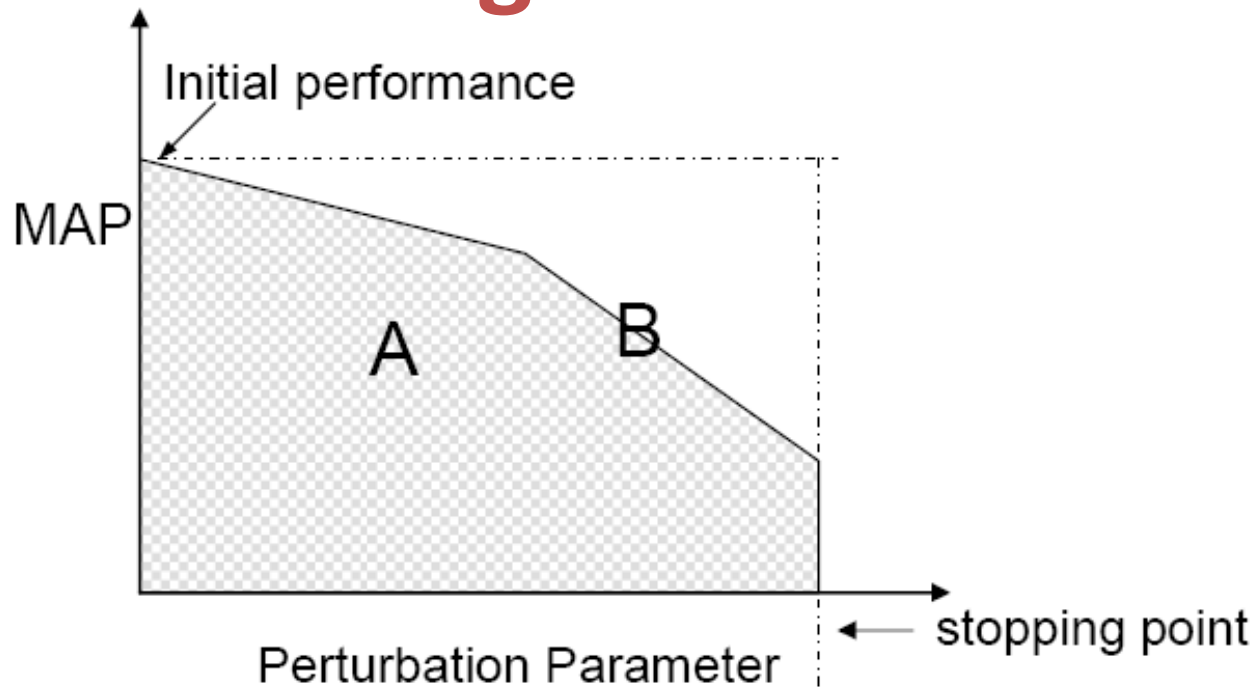
Relevance Preserving Perturbations

Name	Semantic	Perturbation
relevance addition	Add a query term to a relevant document	$aT(t, D, K)$, where $D \in \mathcal{D}_r, t \in Q$
noise addition	Add a noisy term to a document	$aT(t, D, K)$, where $D \in \mathcal{D}, t \in \mathcal{V}_n$
internal term growth	Add a term to a document that originally contains the term	$aT(t, D, K)$, where $D \in \mathcal{D}, t \in D$
document scaling	Concatenate D with itself K times	$cD(D, D, K)$, where $D \in \mathcal{D}$
relevant doc. concatenation	Concatenate two relevant documents K times	$cD(D_1, D_2, K)$, where $D_1, D_2 \in \mathcal{D}_r$
non-relevant doc. concatenation	Concatenate two non-relevant documents K times	$cD(D_1, D_2, K)$, where $D_1, D_2 \in \mathcal{D}_n$
noise deletion	Delete a term from a non-relevant document	$dT(t, D, K)$, where $D \in \mathcal{D}_n, t \in D$
document addition	Add a document to the collection	$aD(D, K)$
document deletion	Delete a document from the collection	$dT(D)$, where $D \in \mathcal{D}$.

Diagnostic Tests - Procedure

1. Identify some desirable properties of a reasonable retrieval function for diagnosis
2. Connect the properties that are to be diagnosed with relevance-preserving perturbations
3. Use the perturbation parameter of the operators to control the degree of perturbation
4. Record the empirical performance of retrieval functions on the corresponding perturbed collections, and stop doing this when you get enough information

Example Curve for the Results of a Diagnostic Test



$$\text{Performance Ratio} = \frac{\text{area_under_curve}}{\text{area_under_line_through_init_point}}$$

- Flat curve: the function being tested is completely insensitive to such perturbation
- Dropping curve: the function suffered from the perturbation

Length Variation Sensitivity Tests

- Length variance reduction test (LV1):

Use the document scaling perturbation to make all the documents to have similar or identical length, and thus reduce the variance of document lengths:

$$\forall D \in \mathcal{D}, cD(D, D, K), K = \frac{(1 - \beta) \times |D| + \beta \times 1000000}{|D|}, 0 \leq \beta \leq 1$$

- Length variance amplification test (LV2):

Amplify the differences in document lengths and make distribution of document lengths skewer:

$$\forall D \in \mathcal{D}, cD(D, D, K), K = |D| \times \beta$$

Length Variation Sensitivity Tests (cont'd)

- Length scaling test (LV3):
Concatenate all the documents with themselves K times, where K is same for all the documents. In this way, the length variance would change but the relative length ratio remains the same:

$$\forall D \in D, cD(D, D, K), K = \text{const}$$

Term Noise Resistance Tests

- Noise addition test (TN):

Add noise (i.e., non-relevant terms) to documents.

$$\forall D \in \mathcal{D}, aT(t_n, D, K), t_n \in V_n$$

1. Constant growth: K is a constant, add the same number of noisy terms to all documents
2. Linear growth: $K = \beta \times |D| - \sum_{t \in Q} c(t, D)$, $\beta > 1$
the number of added noisy terms is linear to the original document length.

TF-LN Balance Tests

Examine the ability of a retrieval function to balance TF and LN. The main idea is to increase the occurrences of the query terms that are already in the documents.

- Single query term growth test (TG1):

Increase the term occurrence for only one random query term:

t : random query term, $\forall D \in \mathcal{D}$, if $t \in D$, perform $aT(t, D, K)$

- Majority query term growth test (TG2):

Increase the term occurrences for all but one random query term:

t : random query term, $\forall D \in \mathcal{D}$, $\forall t' \in \mathcal{Q} - \{t\}$, if $t' \in D$, perform $aT(t', D, K)$

- All query term growth test (TG3):

Perform the internal term growth perturbation for all query terms:

$\forall D \in \mathcal{D}$, $\forall t \in \mathcal{Q}$, if $t \in D$, perform $aT(t, D, K)$

Tests and Corresponding Interpretations

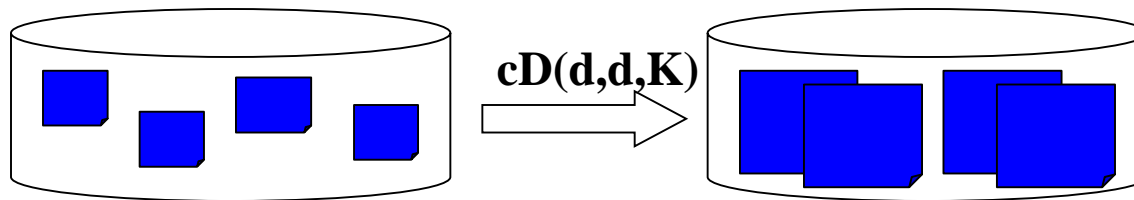
Test	Interpretation of higher value	Tag
length variance reduction	have less gain on length normalization	LV1
length variance amplification	be more robust to larger document variance	LV2
length scaling	better at avoiding over-penalizing long documents	LV3
term noise addition	penalize long documents more appropriately	TN
single query term growth	favor documents with more distinct query terms	TG1
majority query term growth	favor documents with more query terms	TG2
all query term growth	balance TF and LN more appropriately	TG3

Length Scaling Test

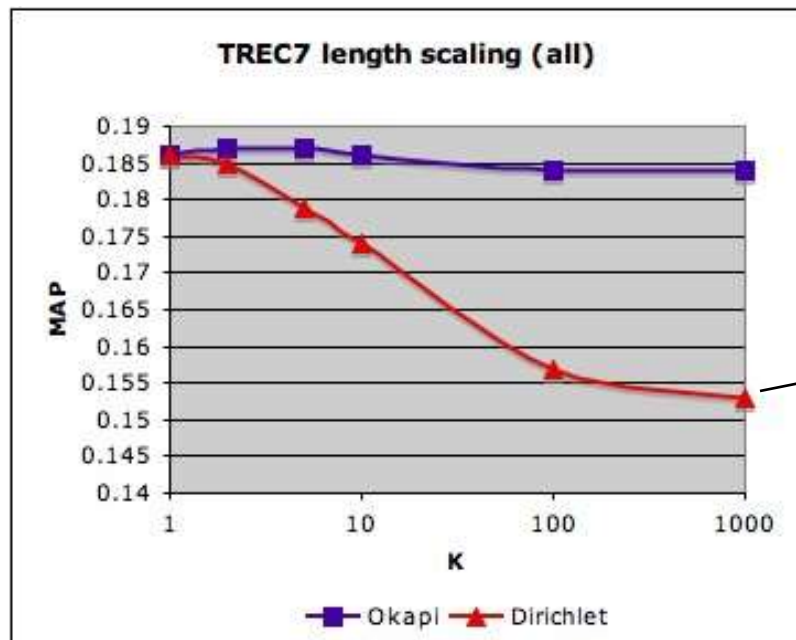
1. Identify the aspect to be diagnosed

test whether a retrieval function over-penalizes long documents

2. Choose appropriate relevance-preserving perturbations



3. Perform the test and interpret the results



Dirichlet over-penalizes long documents!

Identifying the weaknesses makes it possible to improve the performance

	trec8			wt2g			fr88-89		
	MAP	#RRel	P@20	MAP	#RRel	P@20	MAP	#RRel	P@20
Dir.	0.257	2838	0.397	0.302	1875	0.372	0.207	741	0.185
M.D.	0.262	2874	0.415	0.321	1930	0.395	0.224	811	0.191
Piv.	0.244	2826	0.402	0.288	1924	0.369	0.223	822	0.206
M.P.	0.256	2848	0.411	0.316	1940	0.392	0.230	867	0.202

Questions?