



Efficient graph-based context-sensitive search

Carlos Castillo¹ Debra Donato¹ Aris Gionis¹
Antti Ukkonen²

¹Yahoo! Research Barcelona – Catalunya, Spain

²Helsinki University of Technology – Helsinki, Finland

Searching and Ranking in Structured Text Repositories

June 27th, 2008

University of Twente

Enschede, The Netherlands

Outline

- 1 Introduction
- 2 Overview of the Approach
- 3 Ranking algorithm
- 4 Features
- 5 Experiments

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

- 1 Introduction
- 2 Overview of the Approach
- 3 Ranking algorithm
- 4 Features
- 5 Experiments

Keyword-based Search

$$q = \langle t_1, t_2, \dots, t_n \rangle$$

Goal: finding a ranked list of documents, from a given document collection, that are the most relevant to the query

- queries consist of very few terms (< 4);
- queries are often ambiguous and provide incomplete information;
- users information needs are difficult to be accessed;
- need of additional source of information \rightarrow *context*.

Motivation

Keyword-based Search

$$q = \langle t_1, t_2, \dots, t_n \rangle$$

Goal: finding a ranked list of documents, from a given document collection, that are the most relevant to the query

- queries consist of very few terms (< 4);
- queries are often ambiguous and provide incomplete information;
- users information needs are difficult to be accessed;
- need of additional source of information \rightarrow *context*.

Motivation

Keyword-based Search

$$q = \langle t_1, t_2, \dots, t_n \rangle$$

Goal: finding a ranked list of documents, from a given document collection, that are the most relevant to the query

- queries consist of very few terms (< 4);
- queries are often ambiguous and provide incomplete information;
- users information needs are difficult to be accessed;
- need of additional source of information \rightarrow *context*.

Motivation

Keyword-based Search

$$q = \langle t_1, t_2, \dots, t_n \rangle$$

Goal: finding a ranked list of documents, from a given document collection, that are the most relevant to the query

- queries consist of very few terms (< 4);
- queries are often ambiguous and provide incomplete information;
- users information needs are difficult to be accessed;
- need of additional source of information \rightarrow *context*.

Motivation

Keyword-based Search

$$q = \langle t_1, t_2, \dots, t_n \rangle$$

Goal: finding a ranked list of documents, from a given document collection, that are the most relevant to the query

- queries consist of very few terms (< 4);
- queries are often ambiguous and provide incomplete information;
- users information needs are difficult to be accessed;
- need of additional source of information \rightarrow *context*.

Motivation

Keyword-based Search

$$q = \langle t_1, t_2, \dots, t_n \rangle$$

Goal: finding a ranked list of documents, from a given document collection, that are the most relevant to the query

- queries consist of very few terms (< 4);
- queries are often ambiguous and provide incomplete information;
- users information needs are difficult to be accessed;
- need of additional source of information \rightarrow *context*.

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

Context-Sensitive Search

$\langle C, q \rangle$

Goal: increasing relevance.

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

Context-Sensitive Search

$$\langle C, q \rangle = \langle C, t_1, t_2, \dots, t_n \rangle$$

Goal: increasing relevance.

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

Context-Sensitive Search

$$\langle C, q \rangle = \langle C, t_1, t_2, \dots, t_n \rangle$$

Goal: increasing relevance.

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

Context-Sensitive Search

$$\langle C, q \rangle = \langle C, t_1, t_2, \dots, t_n \rangle$$

Goal: increasing relevance.

Context in XML Search and Ranking [Botev and Shanmugasundaram, 2005]

Large XML Library ranging from Shakespeare's plays to scientific papers

- Context (user's interest): Shakespeare's plays
- $q = \langle \text{speech AND process} \rangle$
- XPath query: `//Play[author='William Shakespeare']`

Goal: limiting the scope of her search.

Context in XML Search and Ranking [Botev and Shanmugasundaram, 2005]

Large XML Library ranging from Shakespeare's plays to scientific papers

- Context (user's interest): Shakespeare's plays
- $q = \langle \text{speech AND process} \rangle$
- XPath query: `//Play[author='William Shakespeare']`

Goal: limiting the scope of her search.

Context in XML Search and Ranking [Botev and Shanmugasundaram, 2005]

Large XML Library ranging from Shakespeare's plays to scientific papers

- Context (user's interest): Shakespeare's plays
- $q = \langle \text{speech AND process} \rangle$
- XPath query: `//Play[author='William Shakespeare']`

Goal: limiting the scope of her search.

Context in XML Search and Ranking [Botev and Shanmugasundaram, 2005]

Large XML Library ranging from Shakespeare's plays to scientific papers

- Context (user's interest): Shakespeare's plays
- $q = \langle \text{speech AND process} \rangle$
- XPath query: `//Play[author='William Shakespeare']`

Goal: limiting the scope of her search.

Context in Graph-based Frameworks

Context-sensitive search in Wikipedia

Same model: $\langle C, q \rangle = \langle C, t_1, t_2, \dots, t_n \rangle$

- q : information that users is interested in finding
- C : source Wikipedia document

The context could be used for disambiguation: if $q = \textit{apple}$

- $C = \text{en.wikipedia.org/wiki/Fruit} \rightarrow \text{en.wikipedia.org/wiki/Apple}$;
- $C = \text{www.dell.com} \rightarrow \text{www.apple.com}$

Context in Graph-based Frameworks

Context-sensitive search in Wikipedia

Same model: $\langle C, q \rangle = \langle C, t_1, t_2, \dots, t_n \rangle$

- q : information that users is interested in finding
- C : source Wikipedia document

The context could be used for disambiguation: if $q = \textit{apple}$

- $C = \text{en.wikipedia.org/wiki/Fruit} \rightarrow \text{en.wikipedia.org/wiki/Apple}$;
- $C = \text{www.dell.com} \rightarrow \text{www.apple.com}$

Context in Graph-based Frameworks

Context-sensitive search in Wikipedia

Same model: $\langle C, q \rangle = \langle C, t_1, t_2, \dots, t_n \rangle$

- q : information that users is interested in finding
- C : source Wikipedia document

The context could be used for disambiguation: if $q = \textit{apple}$

- $C = \text{en.wikipedia.org/wiki/Fruit} \rightarrow \text{en.wikipedia.org/wiki/Apple}$;
- $C = \text{www.dell.com} \rightarrow \text{www.apple.com}$

Example: Search while browsing

User reads a particular page and he wants to find more information about a particular term or phrase in the page.

- Context: article about Euler
- $q = \langle \text{problem of seven bridges} \rangle$
- Context: Guns N Roses
- $q = \langle \text{chili peppers} \rangle$

Example: Search while browsing

User reads a particular page and he wants to find more information about a particular term or phrase in the page.

- Context: article about Euler
- $q = \langle \text{problem of seven bridges} \rangle$
- Context: Guns N Roses
- $q = \langle \text{chili peppers} \rangle$

Example: Search while browsing

User reads a particular page and he wants to find more information about a particular term or phrase in the page.

- Context: article about Euler
- $q = \langle \text{problem of seven bridges} \rangle$
- Context: Guns N Roses
- $q = \langle \text{chili peppers} \rangle$

Example: Link finding

A Wikipedia editor is revising an existing document and he wants to quickly find a finding of a good link to a document that provides a description of a term or a phrase

- Context: article editor is editing
- $q = \langle \text{phrase to be linked} \rangle$

Example: Link finding

A Wikipedia editor is revising an existing document and he wants to quickly find a finding of a good link to a document that provides a description of a term or a phrase

- Context: article editor is editing
- $q = \langle \text{phrase to be linked} \rangle$

Challenges

1st Challenge

Efficiently find search results in the search context without having to touch irrelevant content

2nd Challenge

Effectively rank keyword search query evaluated in a search context

Extensions of the model

The context-sensitive problem has many other natural application:

- already used in Yahoo! News (`news.yahoo.com`)
- potential application: Yahoo! answers (`answers.yahoo.com`)

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

- 1 Introduction
- 2 Overview of the Approach**
- 3 Ranking algorithm
- 4 Features
- 5 Experiments

Framework

- $G = (V, E)$
 - V nodes with text information;
 - E hyperlinks (?)
- query: $\langle q, p \rangle$
 - $q = \langle t_1, t_2, \dots, t_n \rangle$
 - $p \in V$
- task: finding nodes $\{x\} \in V$ relevant to the query pair $\langle q, p \rangle$

Framework

- $G = (V, E)$
 - V nodes with text information;
 - E hyperlinks (?)
- query: $\langle q, p \rangle$
 - $q = \langle t_1, t_2, \dots, t_n \rangle$
 - $p \in V$
- task: finding nodes $\{x\} \in V$ relevant to the query pair $\langle q, p \rangle$

Framework

- $G = (V, E)$
 - V nodes with text information;
 - E hyperlinks (?)
- query: $\langle q, p \rangle$
 - $q = \langle t_1, t_2, \dots, t_n \rangle$
 - $p \in V$
- task: finding nodes $\{x\} \in V$ relevant to the query pair $\langle q, p \rangle$

Framework

- $G = (V, E)$
 - V nodes with text information;
 - E hyperlinks (?)
- query: $\langle q, p \rangle$
 - $q = \langle t_1, t_2, \dots, t_n \rangle$
 - $p \in V$
- task: finding nodes $\{x\} \in V$ relevant to the query pair $\langle q, p \rangle$

Graph-Theoretical Framework

Query node places the query in a particular position of the graph, and nodes in the neighborhood of the graph are better-suited answers.

- Graph-based features (in-degree, PageRank, graph-based distance and spectral distance between the source and the target node.
- Inspired by Personalized PageRank [Haveliwala, 2002]
- Given $\langle q, p \rangle$, the main idea is performing PPR on G :
 - $\alpha \rightarrow$ random walk;
 - $1 - \alpha \rightarrow$ return back to page p .

Efficiency Issues

Scalability

time: unfeasible the computation at query time

space: one vector for each node.

Solutions

- Page-specific PageRank with jumps to a smaller number of landmarks.
- Clustering of the graph and computation of one single cluster-specific PageRank for all pages in the cluster.

Set of features for ranking

Task: to rank the nodes of G in decreasing order of their relevance to the query $\langle q, p \rangle$

Basic approach: to compute a score for each node $x \in V$, so that the score measures the relevance of the node x with respect to the query $\langle q, p \rangle$

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G .
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x .
 - indegree;
 - PageRank;

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G .
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x .
 - indegree;
 - PageRank;

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x .
 - indegree;
 - PageRank;

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x
 - indegree;
 - PageRank;

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x
 - indegree;
 - PageRank;

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x
 - indegree;
 - PageRank;

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x
 - indegree;
 - PageRank;

Set of features for ranking

Features

- Features that measure the relevance of the text d_x of the node x with respect to the terms of the query q .
 - Traditional IR and Web search: TF/IDF, BM25, etc.
- Features that measure the similarity between the text d_x of the node x and the text d of the query page p .
 - cosine similarity measures;
 - Jaccard similarity;
- Features that measure the similarity between the nodes x and p with respect to their position in the graph G
 - length of the shortest path from p to x ;
 - number of common successors and predecessors;
 - Page-specific PageRank with teleport vector to the source page
- Query-independent graph-based features of the node x
 - indegree;
 - PageRank;

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

- 1 Introduction
- 2 Overview of the Approach
- 3 Ranking algorithm**
- 4 Features
- 5 Experiments

Ranking Algorithm

Machine-learning approach:

features + ground-truth information

- Given $\langle q, p \rangle$ and a candidate set $C \subseteq V$, compute for each $u \in C$ the feature vector $\mathbf{x}_{\langle q, p \rangle}^u$.
- Final score of u is given by a ranking function $\mathbf{w}^T \mathbf{x}_{\langle q, p \rangle}^u$, where \mathbf{w} is a vector of weights.
- RankSVM [Joachims, 2002] to compute an optimal \mathbf{w} .

Candidate Set

Classical text-based IR approaches do not take the structure into account

- documents ranked with respect their global relevance;
- for rare ambiguous terms, many of the relevant pages might be missed:
 - query: *strawberry* → plant or fruit;
 - query: *strawberry* + context: *The Beatles* → *Strawberry Fields Forever*

Solution: Pruning unwanted documents from the candidate set

Candidate Set

Classical text-based IR approaches do not take the structure into account

- documents ranked with respect their global relevance;
- for rare ambiguous terms, many of the relevant pages might be missed:
 - query: *strawberry* → plant or fruit;
 - query: *strawberry* + context: *The Beatles* → *Strawberry Fields Forever*

Solution: Pruning unwanted documents from the candidate set

Candidate Set

Classical text-based IR approaches do not take the structure into account

- documents ranked with respect their global relevance;
- for rare ambiguous terms, many of the relevant pages might be missed:
 - query: *strawberry* → plant or fruit;
 - query: *strawberry* + context: *The Beatles* → *Strawberry Fields Forever*

Solution: Pruning unwanted documents from the candidate set

Procedure for Computing the Candidate Set

Observation

Most of the “correct” target nodes to many queries are within short distance from the query node. The underlying intuition is that documents corresponding to nodes close to p , in terms of graph distance, are more likely to be better answers to an ambiguous query.

- query: *bach* + context *classical music* → composer *J. S. Bach*;
- query: *bach* + context: *literature* → *Richard Bach*.

Procedure for Computing the Candidate Set

Observation

Most of the “correct” target nodes to many queries are within short distance from the query node. The underlying intuition is that documents corresponding to nodes close to p , in terms of graph distance, are more likely to be better answers to an ambiguous query.

- query: *bach* + context *classical music* → composer *J. S. Bach*;
- query: *bach* + context: *literature* → *Richard Bach*.

Procedure for Computing the Candidate Set

Procedure

- 1 Use a pre-computed inverted index to get all documents that contain all query terms.
- 2 Run a breadth-first search (BFS) starting from the query node following the forward links up to depth h .
- 3 Let C contain all nodes that are in the intersection of the documents containing q and nodes visited by BFS.

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

- 1 Introduction
- 2 Overview of the Approach
- 3 Ranking algorithm
- 4 Features**
- 5 Experiments

Content-based features

BM25

BM25:

$$\sum_{i=1}^m \text{IDF}(q_i) \cdot \frac{f(q_i, d)(k_1 + 1)}{f(q_i, d) + k_1(1 - b + b|d|/\bar{d})},$$

where $\text{IDF}(q_i)$ is the *inverse document frequency* of the query term q_i , typically defined as

$$\text{IDF}(q_i) = \log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}.$$

where $N = |V|$, $k_1 = 1.2$ and $b = 0.75$.

Content-based features

Textual similarity

Textual similarity: Let $T(d_v)$ be the set of terms contained in document d_v . The textual similarity is defined as

$$J(T(d_u), T(d_v)) = \frac{|T(d_u) \cap T(d_v)|}{|T(d_u) \cup T(d_v)|}.$$

Jaccard coefficient

Ranking Features

Link-based features

Predecessor similarity : Let $P(v) = \{u \in V | (u, v) \in E\}$
the predecessor similarity (PRED) is
 $J(P(u), P(v))$.

Successor similarity : Let $S(v) = \{u \in V | (v, u) \in E\}$
the successor similarity (SUCC) is $J(S(u), S(v))$.

Ranking Features

Link-based features

Spectral Distance : an embedding $\phi : V \rightarrow \mathbb{R}^m$ of the graph to a low dimensional Euclidean space and consider the distances of the nodes in this space.

- 1 Laplacian matrix: $\mathcal{L}_G = D - A$, where D is a diagonal matrix having $A_{ii} = d_i$.
- 2 projection $\phi : V \rightarrow \mathbb{R}^m$ ($(m + 1)$ eigenvectors of \mathcal{L}_G)
- 3 The spectral distance (SD) is the Euclidean distance between $\phi(u)$ and $\phi(v)$.

Property: *near-by* nodes u and v in the graph G the distance between the vectors $\phi(u)$ and $\phi(v)$ is short.

Ranking Function

Given the query $\langle q, p \rangle$, we define the score of a node u as

$$\text{score}(u) = \mathbf{w}^T \mathbf{x}_{\langle q, p \rangle}^u.$$

We use RankSVM [Joachims, 2002] is a supervised method that solves this optimization problem.

$$\frac{1}{2} \mathbf{w}^T \mathbf{w} + c \sum \xi_{u_i, v_j}$$

is minimized, subject to the constraints

$$\forall (u_i, v_j) \in P : \mathbf{w}^T \mathbf{x}_{\langle q, p \rangle}^{u_i} > \mathbf{w}^T \mathbf{x}_{\langle q, p \rangle}^{v_j} + 1 - \xi_{u_i, v_j},$$

where all slack-variables ξ_{u_i, v_j} are non-negative.

Training set: ambiguous queries in Wikipedia

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

- 1 Introduction
- 2 Overview of the Approach
- 3 Ranking algorithm
- 4 Features
- 5 Experiments**

Evaluation

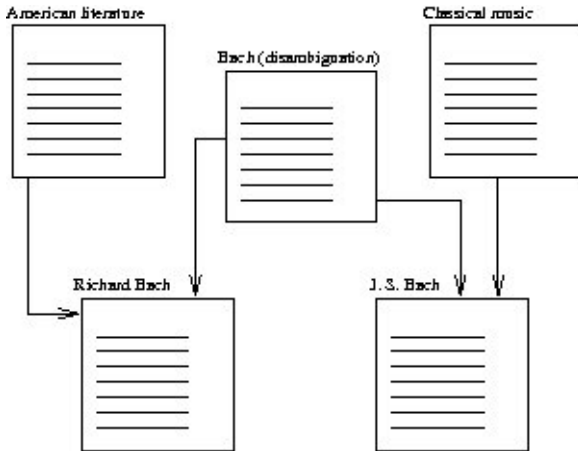


Figure: An illustration of the use of Wikipedia disambiguation pages for finding ambiguous query terms (“bach”) for different contexts (“American literature” or “Classical music”).

Individual features without BFS

	success rate @ k			mean	median
	k=1	k=5	k=10		
BM25	0	12	25	285	38
TEXT	0	21	28	3435	121
SUCC	0	21	30	1634	56
PRED	0	23	34	1308	63
SD	1	12	19	739	105

Table: Results for individual features when the candidate set contains all documents that match the query terms.

Individual features with BFS

	success rate @ k			mean	median
	k=1	k=5	k=10		
BM25	9	49	61	15.6	5
TEXT	7	23	27	919	58
SUCC	8	22	31	465	31
PRED	8	26	32	232	27
SD	6	19	27	188	49

Table: Results for individual features when the candidate set contains documents that match the query terms and are found by a BFS traversal to depth 3 starting from the query node.

Context-sensitive search — Results

	success rate @ k			mean	median
	k=1	k=5	k=10		
BFS_TPR	44	80	83	2.1	1
BFS_CPR	23	56	71	11.6	3
BFS_LPR	20	55	67	14.3	4
BFS_NPR	25	59	66	16.7	3
BFS_GBF	7	27	33	152	25
BFS_CBF	20	52	62	67.2	3
NOBFS_TPR	2	92	99	3.1	2
NOBFS_CPR	0	55	72	49.2	5
NOBFS_LPR	0	50	63	66.1	6
NOBFS_NPR	0	59	67	81.7	5
NOBFS_GBF	0	25	35	451	31

Table: Using RankSVM with different combinations of features. BFS and NOBFS tell if the candidate set was pruned by a BFS search starting from the query node. TPR, CPR and LPR stand for “true”, “cluster” and “landmark” pageranks, respectively. GBF (CBF) means that only the graph-based (content-based) features were in use.

Qualitative results: “Jaguar”

Context 1 Animal	Context 2 Computer Science	Context 3 Automobile
Jaguar	Atari Jaguar	Jaguar XJ
Jaguar (car)	Mac OS X v10.2	Jaguar (car)
European Jaguar	Daimler Motor Company	Jaguar XK
Fender Jaguar	SEPECAT Jaguar	Jaguar XJS
Black panther	Mac OS X	Jaguar S-Type
Fender Bass VI	Jaguar X-Type	Jaguar E-Type
Pantherinae	Polymorphism (biology)	Jaguar X-Type
Panthera	British Leyland Motor Corp.	Jaguar XJ220
Jaguar XJ220	Royal Air Force	Daimler Motor Comp.
South America	HMS Jaguar (F34)	Jaguar AJ-V8 engine

Qualitative results: “Cuckoo”

Context 1 Animal	Context 2 Computer Science	Context 3 Film
Cuckoo	Sopwith Cuckoo	One Flew over ...
Common Cuckoo	The Cuckoo's Egg	Jack Nicholson
Cuculus	Robert Morris (cryptogr.)	Ken Kesey
Coccyzus	Oberon-2	Academy Award
Yellow Cuckoo	Clifford Stoll	Louise Fletcher
Sumatran cuckoo	Bernardo Pasquini	Cuckoo clock
Clamator	Dictionary attack	1975 in film
Striped Cuckoo	Mike Muuss	1970s
Dideric Cuckoo	Computer insecurity	48th Academy Award
Crotophagidae	Louise Fletcher	Milos Forman

Qualitative results: “Watson”

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

Context 1

Behaviorism

John B. Watson
Radical behaviorism
Doctor Watson
Little Albert experiment
Classical conditioning
James Watson
James Cronin
Robert Watson-Watt
The Advent. of the Dying Detective
William Watson

Context 2

DNA

James D. Watson
Francis Crick
Molecular structure of Nucleic Acids
Rosalind Franklin
Maurice Wilkins
History of molecular biology
Watson and Crick
The Double Helix
James Watson
Gene

Context 3

Mystery

Doctor Watson
David Burke
Without a Clue
Michael Mallory
Muse Watson
The Doctor's Case
James Cronin
Nigel Bruce
The Italian Secretary
The Seven-Per-Cent Solution

Efficient
graph-based
context-sensitive
search

Castillo,
Donato, Gionis,
Ukkonen

Introduction

Overview of the
Approach

Ranking
algorithm

Features

Experiments

- 1 Introduction
- 2 Overview of the Approach
- 3 Ranking algorithm
- 4 Features
- 5 Experiments



Botev, C. and Shanmugasundaram, J. (2005).
Context-sensitive keyword search and ranking for xml.
In 8th International Workshop on the Web and Databases (WebDB).



Haveliwala, T. (2002).
Topic-sensitive pagerank.
In Proceedings of the 11th WWW Conference.



Joachims, T. (2002).
Optimizing search engines using clickthrough data.
In Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD).