Effective Structured Query Formulation for Session Search

Dongyi Guan, Grace Hui Yang, Nazli Goharian

Speaker: Grace Hui Yang

Department of Computer Science
Georgetown University

Nov 9, 2012@TREC

1

Introduction

- A session contains
  - Interactions
    - Previous Queries $q_1, q_2, \dots, q_{n-1}$
    - Previous Search Results
    - Click Information
  - Current Query q_n
- A *retrieval task*
  - So, we basically use Lemur – a strong baseline
- The problem becomes
  - how to make good use of Lemur
  - … and how to improve over Lemur

2

Outline

- Our Approach
  - Structured Query Formulation
  - Query Expansion
  - Duplicated Queries
  - Document Re-ranking by Dwell Time
- Experiments

3

Structured Query Formulation

- *We emphasize on formulating effective Lemur queries*
  - *then submit the queries to Lemur*
- An observation:
  - In a query, several words often bundle together as a phrase to express a coherent meaning; we call them ***nuggets***
  - *Nuggets* are substrings in q_n, similar to phrases but not necessarily as semantically coherent as phrases
  - Using nuggets to form structured query could be more effective than using plain free text query

servering spinal cord paralysis ⇨ servering #1(spinal cord) paralysis

word nugget word

4

How to Identify Nuggets

- More Observations:
  - *a valid nugget (within a query) appears frequently in the top returned snippets for the query*
  - *Even if the words in a nugget do not appear continuously in the snippets, they appear close.*

> ...preservation uk **spinal cord** injury care in egypt cky **cord**
>
> noats guitar viewsonic power cord malfunction **spinal cord**
>
> stimulator...injury **spinal cord** dell...extension **cord**
>
> nylon...**cord paralysis** vocal vegas…

A sample snippet for TREC 2012 session 53 query **servering spinal cord paralysis**, where "spinal cord" are grouped as a nugget #1(spinal cord)

5

How to Identify Nuggets

1. Send q_n to Lemur, get initial retrieval results
2. look for possible nuggets in the top k snippets
   - High frequency adjacent words
   - Other words which frequently co-occur within a certain proximity

6

Strict Method

$q = w_1 w_2 \cdots w_l$

$$\frac{\text{count}(w_i w_{i+1};\text{Snippet})}{\min(\text{count}(w_i w_{i+1};\text{Snippet}),\text{count}(w_i w_{i+1};\text{Snippet}))} \geq \theta \quad \Rightarrow \quad w_i w_{i+1} \text{ are connected}$$

$nugget = \#1(w_i w_{i+1} \cdots w_{i+k})$ $\quad w_i w_{i+1} \cdots w_{i+k}$ are connected

Example

Query

servering spinal cord paralysis

Snippet

...preservation uk **spinal cord** injury care in egypt cky **cord** noats guitar viewsonic power cord malfunction **spinal cord** stimulator...injury **spinal cord** dell...extension **cord** nylon...**cord paralysis** vocal vegas…

Structured Query

servering #1(spinal cord) paralysis

RL1 Query

7

Relaxed Method

Define centroid of a word w_i is $\bar{x}(w_i) = \frac{1}{k} \cdot \sum_{t=1}^{k} \frac{\sum_j x_j(w_i; S_t)}{\text{count}(w_i; S_t)}$

Predict the nugget window size $nugget = \begin{cases} \#1(w_i\ w_{i+1}) & |\bar{x}(w_i) - \bar{x}(w_{i+1})| \leq 5 \\ \#2(w_i\ w_{i+1}) & 5 < |\bar{x}(w_i) - \bar{x}(w_{i+1})| \leq 10 \\ \phi & |\bar{x}(w_i) - \bar{x}(w_{i+1})| > 10 \end{cases}$

Example

Query

marsupial cartoon character

Structured Query

marsupial #2(cartoon character)

Snippet

…about a **cartoon character**. For the carnivorous **marsupial**, see Tasmanian…animated
7 8 **9** **10** 11 12 13 **14** 15 16 23
cartoon character in the…series of **cartoons**…the **character** after…between the
24 **25** 26 27 32 33 **34** 38 **39** 40 55 56
marsupial…encyclopedia **cartoon character**) Jump…propelled the **character** to new…
57 68 **69** **70** 71 80 81 **82** 83 84
animated **cartoon character**…Tunes series of **cartoons**. The **character** appeared in…
89 **90** **91** 98 99 100 **101** 102 **103** 104 105

$$\left.\begin{array}{l} \bar{x}(\text{marsupial}) = 35 \\ \bar{x}(\text{cartoon}) = 54 \\ \bar{x}(\text{character}) = 60 \end{array}\right\} \Rightarrow \left\{\begin{array}{l} \text{marsupial} \\ \#2(\text{cartoon character}) \end{array}\right.$$

RL1 Query

8

Query Expansion with Previous Queries

1. Extract nuggets and words from every query $q_1, q_2, \ldots, q_n$ in a session
2. Combine them and weigh them by per-query weight λ_k

#weight(
λ_1 #combine($nugget_{11}\ nugget_{12} \cdots nugget_{1m}\ w_{11}\ w_{12} \cdots w_{1r}$)
λ_2 #combine($nugget_{21}\ nugget_{22} \cdots nugget_{2m}\ w_{21}\ w_{22} \cdots w_{2r}$)
…
λ_n #combine($nugget_{n1}\ nugget_{n2} \cdots nugget_{nm}\ w_{n1}\ w_{n2} \cdots w_{nr}$)
)

RL2 Query

9

Query Expansion with previous queries

Weighting Schemes

- Uniform
  All queries are assigned the same weight. $\lambda_k = 1$
- Previous vs. current
  All previous queries share the same weight while the current query uses a complementary and higher weight

$$\lambda_k = \begin{cases} \lambda_p & k = 1,2,\cdots n-1 \\ 1-\lambda_p & k = n \end{cases} \qquad \lambda_p = 0.4$$

- Distance-based
  The weights are distributed based on how far a query's position in the session is from the current query

$$\lambda_k = \begin{cases} \dfrac{\lambda_p}{n-k} & k = 1,2,\cdots n-1 \\ 1-\lambda_p & k = n \end{cases} \qquad \lambda_p = 0.4$$

10

Query Expansion with search results

Anchor Log

- Collected by *harvestlink* in the Lemur toolkit
- Extract the top 5 frequent anchor text in the previous results
- Weights are proportional to normalized frequency of anchor text

$$
\begin{aligned}
&\#\text{weight}(\\
&\lambda_1\ \#\text{combine}(nugget_{11}\ nugget_{12}\cdots nugget_{1m}\ w_{11}\ w_{12}\cdots w_{1r}) \\
&\lambda_2\ \#\text{combine}(nugget_{21}\ nugget_{22}\cdots nugget_{2m}\ w_{21}\ w_{22}\cdots w_{2r}) \\
&\cdots \\
&\lambda_n\ \#\text{combine}(nugget_{n1}\ nugget_{n2}\cdots nugget_{nm}\ w_{n1}\ w_{n2}\cdots w_{nr}) \\
&\beta\omega_1\ \#\text{combine}(e_1)\ \beta\omega_2\ \#\text{combine}(e_2)\cdots\beta\omega_5\ \#\text{combine}(e_5) \\
&)
\end{aligned}
$$

factor

frequency anchor text

RL3/RL4 Query

Example (TREC 2012 session 53)

#weight(1.0 #1(spinal cord) 0.6 consequenses 0.4 paralysis 1.0 servering 0.38 #combine(type of paralysi) 0.0048 #combine(quadriplegia paraplegia) 0.0048 paraplegia 0.0048 #combine(spinal cord injury) 0.0024 #combine(quadriplegic tetraplegic))

11

Duplicated Queries

- Suggest user's intention

1. pocono mountains pennsylvania
2. pocono mountains pennsylvania hotels
3. pocono mountains pennsylvania things to do
4. pocono mountains pennsylvania hotels
5. pocono mountains camelbeach
6. pocono mountains camelbeach hotel
7. pocono mountains chateau resort
8. pocono mountains chateau resort attractions
9. pocono mountains chateau resort getting to
10. chateau resort getting to
11. pocono mountains chateau resort directions

Example: TREC 2012 session 6

12

Duplicated Queries

Assumptions

- If there is a previous query that is the same as the current query q_n, we only use the current query to generate the structured session query
  - The user came back to a previous query, which perhaps indicates that other previous queries are not very satisfying

- If several previous queries (other than q_n) are duplicated, we remove them when formulating the structured session query
  - The user changed the query after checking it twice, which indicates the results of this query is not satisfying

13

Duplicated Queries

Example

TREC 2011 session 22

Used in RL3/RL4

Queries in a Session	Without removing duplicates		Removing duplicates	
	Structured query	nDCG@10	Structured query	nDCG@10
shoulder joint pain	#weight(1.4 joint 0.4 nhs 1.4 pain 1.4 shoulder 0.036 #1(shoulder pain) 0.036 #1(frozen shoulder) 0.01 #1(shoulder pain causes) 0.006 #1(bursitis) 0.006 #1(painful shoulder conditions))	0.5538	#weight(0.9 joint 1.0 pain 1.0 shoulder)	0.6434 (+16.18%)
shoulder joint pain nhs				
shoulder joint pain				

14

Document Re-ranking

Dwell time: the elapsed time that a user stays in the page

$$\Delta t = t_{end} - t_{start}$$

Clicked documents: $\{c_1, c_2, \cdots, c_k\}$

Associated dwell time: $\{\Delta t_1, \Delta t_2, \cdots, \Delta t_k\}$

Re-ranking the returned documents $\{d_j\}$ **by**:

$$s(d_j) = \sum_{i=1}^{k} \mathrm{Sim}(d_j, c_i) \cdot \Delta t_i$$

Using raw dwell time to strongly bias towards SAT (satisfying) clicks

15

Submitted Runs

run	RL1	RL2	RL3	RL4
guphrase1	strict method μ = 4000, k = 10	strict method query expansion μ = 4500, k = 5	strict method query expansion anchor text remove duplicates μ = 4500, k = 5	strict method query expansion anchor text remove duplicates re-ranking by time μ = 4500, k = 5
guphrase2	strict method μ = 3500, k = 10	strict method query expansion μ = 5000, k = 5	strict method query expansion anchor text remove duplicates μ = 5000, k = 5	strict method query expansion anchor text remove duplicates re-ranking by time μ = 5000, k = 5
gurelaxphr	relaxed method μ = 4000, k = 20	relaxed method query expansion μ = 4500, k = 20	relaxed method query expansion anchor text remove duplicates μ = 4500, k = 20	strict method query expansion anchor text remove duplicates re-ranking by time μ = 4500, k = 5

16

Evaluation Results (2012)

nDCG@10 for TREC 2012 runs

run	original	guphrase1	guphrase2	gurelaxphr	Mean of the median
RL1	0.2474	0.2298	0.2265	0.2334	0.1746
RL2		0.2932	0.2839	0.2832	0.1901
RL3		0.3021	0.2995	0.3033	0.216
RL4		0.3021	0.2995	0.29	0.2261

- Terms from previous queries boost the accuracy significantly
  - Big improvement from RL2 to RL1
- Removing duplicated queries improves the search accuracy
  - Improvement from RL3 to RL2
- Grouping terms into nuggets is not effective to 2012 queries
  - Might overfit on 2011 queries

17

Evaluation Results (2011)

nDCG@10 for TREC 2011 RL1 runs. A significant improvement over the baseline is indicated with a † at p<0.05 level and a ‡ at p<0.005 level

Metric	original query	strict	relaxed	2011 Best	2011 Median
nDCG@10	0.3378	**0.3834**	**0.3979**	0.3789	0.3232
%chg		**+13.50%†**	**+17.79%‡**		

- Structured queries built on nuggets improve the accuracy significantly
- Relaxed method even boosts higher
  - allows larger window size, may be more suitable

18

Evaluation Results (2011)

nDCG@10 for TREC 2011 RL2 runs. A significant improvement over the baseline is indicated with a † at p<0.05 level and a ‡ at p<0.005 level

Metric	original query	uniform	previous vs. current	distance-based	2011 Best	2011 Media
nDCG@10	0.3378	**0.4475**	**0.4626**	**0.4431**	0.4281	0.3215
%chg		**32.47%‡**	**36.94%‡**	**31.17%‡**		

- Terms from previous queries significantly boost the search accuracy
- *previous vs. current* outperforms other schemes
  - The intention of user is complicated
  - Cannot assume that the early queries are less important

19

Evaluation Results (2011)

nDCG@10 for TREC 2011 RL3 and RL4 runs. A significant improvement over the baseline is indicated with a † at p<0.05 level and a ‡ at p<0.005 level

Baseline = 0.34 / Method	anchor text				nDCG@10	
	all documents		clicked documents (RL4 only)		Best	Median
	nDCG@10	%chg	nDCG@10	%chg	RL3	RL3
all queries	**0.4695**	38.99%‡	**0.4680**	38.54%‡	0.4307	0.3259
remove duplicated queries	**0.4836**	43.16%‡	**0.4542**	34.46%‡	RL4	RL4
re-rank by dwell time (RL4 only)	0.4435	31.29%‡	N/A		0.4540	0.3354

- Removing duplicated queries improves the accuracy
- Re-ranking does not perform well
  - Ranking by raw dwell time might be rough

20

Conclusions

- Construct structured Lemur queries for session search
- What works:
  - Using previous queries
  - Eliminating duplicates
- What we believe that works
  - Using nuggets to form structured query
    - we did achieve good performance gain over 2011 data
    - … thus keep investigating

21

Thank You

Grace Hui Yang
Department of Computer Science
Georgetown University
huiyang@cs.georgetown.edu

Nov 9, 2012

22