Improving Unsupervised Query Segmentation using Parts-of-Speech Sequence Information

Rishiraj Saha Roy, Yogarshi Vyas and Niloy Ganguly
Indian Institute of Technology Kharagpur
Kharagpur, India - 721302.
{rishiraj, yogarshi.vyas, niloy}@cse.iitkgp.ernet.in

Monojit Choudhury
Microsoft Research India
Bangalore, India - 560001.
monojitc@microsoft.com

ABSTRACT

We present a generic method for augmenting unsupervised query segmentation by incorporating Parts-of-Speech (POS) sequence information to detect meaningful but rare n-grams. Our initial experiments with an existing English POS tagger employing two different POS tagsets and an unsupervised POS induction technique specifically adapted for queries show that POS information can significantly improve query segmentation performance in all these cases.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query formulation

General Terms

Algorithms, Measurement, Experimentation

Keywords

Query segmentation, POS tagging, IR evaluation

1. INTRODUCTION

Query segmentation is the process of breaking down Web search queries into their constituent structural units, thereby aiding the retrieval process [8]. An example of such a segmented query is `arthur conan doyle | short stories | buy online`, where pipes (|) represent segment boundaries. Segmentation is considered as one of the first steps to query understanding, and has attracted a good amount of attention [8, 9, 11]. Unsupervised query segmentation algorithms use statistical word association scores (WAS) [6] to mine potential segments from a corpus, and thus, suffer from the problem of missing out the rarer ones. In this research, we address this problem by applying a novel intuition that POS sequence patterns of frequent segments can be leveraged for extracting the rarer ones that are generally present in the *tail queries* and lack sufficient clickthrough data. Improving performance on tail queries is of great interest to commercial Web search engines.

Our generic strategy to augment WAS-based segmentation techniques with POS information is as follows. First, we construct a lexicon of potential word n-grams from the corpus (say, a query log). This is usually the first step in a WAS-based query segmentation algorithm [9, 12]. Then we identify underlying POS sequences (or POS n-grams) of the lexicon entries, and count their frequency of occurrence. A modified score is then computed for each word n-gram which is a combination of its original WAS and the lexicon frequency of its POS n-gram. New entries are introduced into the lexicon according to this modified score. This process is iterated till convergence of the lexicon. This *augmented lexicon* is used for query segmentation, where the newly derived scores perform the role of the original WAS. This lexicon augmentation is an *offline* process and thus does not add any runtime overhead to the segmentation process.

We conduct experiments using an English POS tagger based on the Penn Treebank (PTB) tagset and a recently proposed compact universal tagset [10]. We also experiment with a tagset that has been induced from the query log in a completely unsupervised fashion [5]. Our results show that POS information from all the three tagsets can lead to significant performance improvement for an unsupervised segmentation algorithm [9].

2. APPROACH

Fig. 1 presents a schematic of the proposed framework to combine WAS and POS information for unsupervised query segmentation. Our method requires a POS tagger for queries, a WAS to be computed from a query log, and a lexicon augmentation scheme.

2.1 POS Tagging

POS tagging is defined as the process of assigning POS labels to the words of a text fragment based on the context. For example, if the input text fragment is `the yellow book`, the corresponding POS labels would be `the`_DT `yellow`_JJ `book`_NN (i.e., Determiner, Adjective and Noun respectively). The framework proposed here is not tied to any specific POS tagging strategy or tagset. To bring out this fact, here we perform initial experiments with two different taggers – the supervised Stanford Log-Linear POS Tagger [14], and a fully unsupervised POS induction technique using graph clustering based on Biemann [5]. The

SIGIR'14, July 6–11, 2014, Gold Coast, Queensland, Australia.

http://dx.doi.org/10.1145/2600428.2609478.

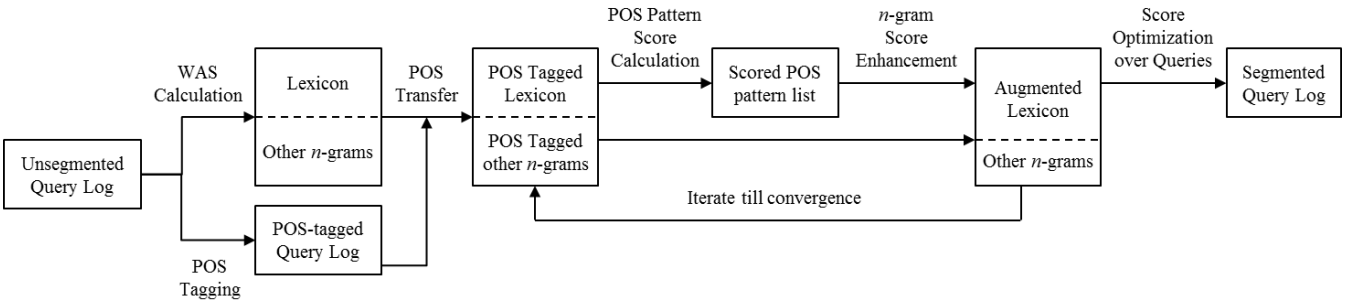

Figure 1: Framework for augmenting unsupervised query segmentation with POS sequence information.

Table 1: Sample clusters produced by Bie-S.

Cluster 1:	bake, casserole, dessert, fry, meatballs, ...
Cluster 2:	athletics, baseball, cycling, football, golf, ...
Cluster 3:	army, citizenship, customs, defence, government, ...
Cluster 4:	battlefield, diablo, godfather, hitman, sims, ...

Stanford Tagger uses the PTB tagset that has 36 tags[1]. Recently, Petrov et al. [10] proposed a universal tagset (UTS) which contains 12 tags and provided a mapping between the PTB (and many other) tags and UTS tags. In order to understand the effect of granularity of the tagset, we also run our experiments for the UTS tagset, which are simply obtained by one-to-one mappings of the PTB tags of the queries labeled by the Stanford Tagger.

Since English Web search queries do not necessarily follow the syntax of the English language, the appropriateness of tagsets such as PTB or UTS for tagging Web search queries is questionable. Therefore, we also experiment with a completely unsupervised POS induction technique based on graph clustering [5] that induces the tagset as well as the tagger from first principles without making any assumptions about the syntactic structure of the language. Moreover, the technique automatically generates the number of tags (clusters). The original method is simplified for queries so as to assign a unique tag to a word (by suitably removing the Viterbi tagging step in [5]), irrespective of the context. This ensures a fast and lightweight tagger that is suitable for a Web search setting. We refer to this tagger (and the associated tagset) as Bie-S (S = Simplified). Table 1 shows parts of sample clusters generated by the Bie-S algorithm on our query log. As we can see, clusters are focused around *topics* like food, sports, governance, and video games. The method resulted in 405 distinct tags.

2.2 Lexicon Augmentation Scheme

Intuition. Traditional unsupervised query segmentation algorithms use a WAS to build a lexicon of meaningful n-grams [9, 12], which is subsequently used to generate the most likely segmentation for a query. Such methods fail to identify rare word n-grams as potential segments. The rarer n-grams, nevertheless, almost always follow the same syntactic structure (or POS sequence pattern) as the frequent ones, and their rarity is by virtue of the rarity of the words rather than the underlying syntactic construction. This fundamental observation led us to the intuition that the WAS of rarer word n-grams could be boosted up if the underlying POS pattern is observed frequently in the set of segments originally extracted by a WAS. In other words, we can learn the common syntactic structures of the segments by extracting statistically significant word co-occurrences, and then in turn, use this knowledge to extract rarer segments. This intuition, which is the primary contribution of this work, is formalized in the following steps.

1. Given a query log Q, the queries are POS tagged using a tagger. Also, a WAS is computed for every unique word n-gram, $\mathbf{w}$, appearing in Q.
2. An initial lexicon L_0 is constructed with the word n-grams (say, `the rolling stones`) that have WAS $\geq \delta$, a user-defined threshold. Let L_i be the lexicon after the i^{th} iteration of the algorithm.
3. Every entry in L_i is assigned a unique POS tag sequence based on how that word n-gram was tagged in Q (say, `the`_DT `rolling`_VBG `stones`_NNS). When the same word n-gram is tagged differently in different queries, we assign the most common POS tags to words in that n-gram sequence as found in Q.
4. For each POS n-gram (or POS pattern) P_j (say, DT-VBG-NNS), we count the number of times P_j appears in L_i. Let us denote this by $count(P_j, i)$.
5. We define a score for P_j as follows:
$$score(P_j, i+1) = score(P_j, i)\ln(e + \alpha e^{-i/\ln(1+count(P_j,i))}) \quad (1)$$
where iteration $i \geq 0$ and α is a tuning parameter. We define $score(P_j, 0) = 1$.
6. The WAS for every unique $\mathbf{w}$ in Q is then combined with its corresponding pattern score as shown in Eq. 2:
$$score(\mathbf{w}, i+1) = score(\mathbf{w}, i) \times score(P_{\mathrm{POS}(\mathbf{w})}, i) \quad (2)$$
where POS($\mathbf{w}$) is the index j of the underlying POS pattern of $\mathbf{w}$. We define $score(\mathbf{w}, 0)$ to be WAS($\mathbf{w}$).
7. L_{i+1} is constructed by including all $\mathbf{w}$ for which $score(\mathbf{w}, i) \geq \delta$.
8. Steps 3 to 8 are repeated till convergence, which we define as a state where $L_i \equiv L_{i-1}$.

The multiplicative factor in Eq. 1 is based on the proximity transformation function used by Tao and Zhai [13], which has all the mathematical characteristics to suit the current

[1] `http://bit.ly/JY5lwb`

Metric	Orig	PTB	UTS	Bie-S
nDCG@5	0.743	$0.751^{\dagger}$	$0.751^{\dagger}$	$0.751^{\dagger}$
nDCG@10	0.747	0.753	0.752	0.752
MAP	0.901	0.905	0.905	0.905
MRR	0.587	0.601	0.598	0.602

Statistically significant improvement over *Orig* marked using † (one-tailed paired t-test, $p < 0.05$).

Table 2: IR performance with different tagsets.

purpose: (a) the value of the function diminishes with each successive iteration, which is necessary because otherwise eventually all n-grams will enter the lexicon; (b) as i grows, this factor approaches unity, which ensures convergence; (c) this factor is proportional to the logarithm of $count(P_j, i)$, which is usually desirable because frequency distributions of n-grams typically follow power laws.

2.3 Segmentation Algorithm

We use the state-of-the-art association score CSR [6] and the related query segmentation algorithm [9]. There are two important novelties in the overall method when applied to the context of queries: (a) A decision is made on the significance of a word n-gram $\mathbf{w}$ only on the basis of the number of queries which contain all the terms of $\mathbf{w}$, thus disallowing frequently misleading unigram statistics to interfere with the decision, and (b) the segmentation procedure is capable of segmenting queries using only a query log, not relying on any other external resource. In our approach, we *do not* use the initial lexicon L_0 to segment queries; rather we use the lexicon $L_{\hat{i}}$ where $\hat{i}$ is the iteration at which convergence occurs. We refer to the segmentation produced using L_0 as the original segmentation *Orig*, over which we aim to improve.

3. EXPERIMENTS

Dataset. For training our system, we use a query log (Q) sampled from Bing Australia in May 2010, consisting of $16.7M$ (M = Million) queries ($11.9M$ distinct). This query log was POS tagged using the Stanford Tagger (using both PTB and UTS tags) as well as the Bie-S algorithm. For evaluating segmentations generated by our approach, we use the dataset released by Saha Roy et al. [11] containing 500 Web search queries[2] with associated URLs and relevance judgments (approx. 30/query, $0-2$ scale, average rating of three annotators). The queries in this dataset are slightly long ($5-8$ words) where segmentation is actually meaningful from an IR perspective. We use the commercially popular open source Apache Lucene[3] to search this collection [11]. Queries 1 to 250 have been used as the development set and 251 to 500 as the test set.

IR-based evaluation. We evaluate our approach using the state-of-the-art IR-based segmentation evaluation framework [11]. The framework uses an oracle-based approach to measure the IR potential of a segmentation algorithm. Several versions of a segmented query are generated by variously quoting the multi-word segments. The performance of the segmentation algorithm, for each query, is assumed to be the performance of the quoted version for the query that retrieves the best results. Table 2 reports

[2] `http://bit.ly/ZS0ybI`

[3] `http://lucene.apache.org/core/`

Table 3: Gaining, unaffected and losing queries.

Tagset	Gain	Same	Loss
PTB	67 (+0.048)	150	33 (−0.060)
UTS	57 (+0.055)	162	31 (−0.068)
Bie-S	67 (+0.042)	140	43 (−0.050)

Numbers in parentheses indicate the average gain/loss in nDCG@10 for each class of queries with respect to the original segmentations.

Tagset	Segmented Query	nDCG@10
Orig	`picture | in | picture | lcd tv`	0.606
PTB, UTS	`picture in picture | lcd tv`	**0.788**
Bie-S	`picture in | picture | lcd tv`	0.747
Orig	`samsung | i900 | omnia | free | games`	0.691
PTB, UTS, Bie-S	`samsung i900 | omnia | free games`	**0.810**
Orig	`richard burns rally | pc | cheats`	0.675
PTB, UTS, Bie-S	`richard burns | rally | pc cheats`	**0.751**

Table 4: Example queries showing IR improvement.

nDCG, MAP and MRR for the original algorithm and the POS-augmented strategy for the three tagsets used, averaged over all the test queries.

All the tagsets result in improvements over the original segmentation, which is statistically significant for nDCG@5. This implies that many better pages are presented in the top-five slots, which is very important for a Web search setting. The improvements, we believe, are because of the meaningful rare n-grams that are discovered by our POS-based method but were originally missed by the WAS alone. At convergence, the PTB, UTS and Bie-S tagsets added $337k$, $447k$ and $452k$ new word n-grams to L_0 respectively. Mean IR performances on the test set for the three tagsets are almost exactly the same (a gain-loss analysis reported later reveals some differences). With respect to this application, the UTS tagset does *not* result in any *loss* of *information* when the 36 PTB tags are collapsed to the 12 UTS tags.

Table 3 reports the numbers of gaining, unaffected and losing queries (in terms of nDCG@10, with respect to the original segmentation without POS information) for each tagset for the optimum α-s. We observe that our method benefits a significant proportion of the queries ($23-27\%$), much higher than the fraction of queries for which the nDCG value drops ($12-17\%$). The three tagsets affect relatively same numbers of queries in all the three ways, even though the number of queries negatively affected is slightly higher for the Bie-S tagset. Since these queries are relatively rare with query frequency between 5 and 15, improvement on a significant fraction of these queries would be of considerable interest to commercial Web search engines. Relative magnitudes of average gains and losses appear comparable.

Table 4 shows representative queries that undergo segmentation change due to the augmented lexicon with a consequent IR benefit. It is evident that all the three tagsets are able to detect relatively rarer n-grams (for example, `picture in picture` and `samsung i900`) which did not feature in the initial lexicon. Our method can also adjust n-gram scores so as to insert new segment boundaries at better locations from an IR perspective (`richard burns rally` → `richard burns | rally`).

Convergence. Even after the convergence of the lexicon, it is possible that the segmentations change over further iterations because the scores of the items in the lexicons can

Tagset	PTB	UTS	Bie-S
Lexicon convergence iteration	**30**	70	**30**
Segmentation convergence iteration	80	**70**	90
Peak IR performance iteration	50	50	**10**
Optimum α	100	**10**	1000

The lowest value in a row is marked in **boldface**.

Table 5: Number of iterations and the optimal α.

continue to change, albeit converging towards specific values. Therefore, we explore an alternative convergence criterion, which is when the segmentation of the queries stabilize for our development set. Nevertheless, we observed that the segmentations so obtained does not necessarily lead to maximum IR performance (say, in terms of nDCG@10). In Table 5 we report the number of iterations required for these two types of convergence – lexicon and segmentation, and also the number of iterations after which peak IR performance was achieved. For all our experiments, the parameter α was tuned on the development set using grid-search for maximizing nDCG@10, and the optimal values for each tagset are also reported in Table 5.

We observe that Bie-S, which is a *deterministic* and hence a *fast* approach, takes only 10 iterations to reach its peak IR performance that is comparable to the nDCG of other tagsets, whereas the other approaches take 50 rounds. This is definitely a big advantage for the unsupervised POS induction approach. For all the tagsets, the nDCG@10 at segmentation convergence is slightly less than the peak value, though this difference is not statistically significant.

Frequent POS patterns. The ten most frequent patterns in the lexicons for the PTB and the UTS tagsets turned out to be NN NN, NN NN NN, JJ NN NN, JJ NN, NN NNS, NN NN NNS, NN IN NN, FW FW, JJ JJ NN, JJ NN NNS, and NOUN NOUN, NOUN NOUN NOUN, ADJ NOUN NOUN, ADJ NOUN, NOUN ADP NOUN, NOUN VERB, NOUN NOUN VERB, VERB NOUN, ADJ ADJ NOUN, NOUN VERB NOUN respectively. The Bie-S tags are system-generated and hence are not readily interpretable.

4. DISCUSSION AND CONCLUSION

We have described some initial experiments for enhancing unsupervised query segmentation using POS sequences, with promising results but scope for improvement. This study connects two orthogonal approaches to segmentation or *chunking* of text fragments – those that rely on purely statistical word association measures [8, 9] and those that try to incorporate linguistic information, used commonly for Natural Language (NL) chunking [1]. While POS tagging of queries have received attention recently [7], and has been used in supervised query segmentation for detecting English noun phrases [2, 3, 4], as far as we know this is the first work that applies POS tagging to unsupervised query segmentation, explores an unsupervised POS induction strategy for queries and proposes a method for combining POS sequence information with WAS. Moreover, unlike past approaches where POS tags are used to identify NL phrases in the queries, our method employs POS sequence information to detect rare n-gram patterns that may or may not correspond to any NL construct, and in fact, can be quite unique to the structure of Web search queries.

This is a work in progress and we plan to expand it along several dimensions. The best strategy for combining WAS with POS pattern counts needs further exploration. It seems that PTB or UTS tagsets are complementary to the Bie-S tagset, and it would therefore be useful to develop techniques that combine these two approaches and exploit the benefits of both. It would be also interesting to study which kind of queries benefit maximally from POS-enhanced segmentation and whether they can be automatically identified. Finally, we believe that there is a huge potential for unsupervised POS induction in query understanding and representation that has not yet been leveraged.

Acknowledgments

The first author was supported by Microsoft Corporation and Microsoft Research India under the Microsoft Research India PhD Fellowship Award.

5. REFERENCES

[1] S. P. Abney. *Parsing by chunks*. Springer, 1992.
[2] M. Bendersky, W. B. Croft, and D. A. Smith. Joint annotation of search queries. In *HLT 2011*, pages 102–111.
[3] M. Bendersky, W. B. Croft, and D. A. Smith. Two-stage query segmentation for information retrieval. In *SIGIR 2009*, pages 810–811.
[4] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *EMNLP-CoNLL 2007*, pages 819–826.
[5] C. Biemann. Unsupervised part-of-speech tagging employing efficient graph clustering. In *COLING-ACL 2006 Student Research Workshop*, pages 7–12.
[6] D. L. Chaudhari, O. P. Damani, and S. Laxman. Lexical co-occurrence, statistical significance, and word association. In *EMNLP 2011*, pages 1058–1068.
[7] K. Ganchev, K. Hall, R. McDonald, and S. Petrov. Using search-logs to improve query tagging. In *ACL 2012*, pages 238–242.
[8] M. Hagen, M. Potthast, A. Beyer, and B. Stein. Towards optimum query segmentation: In doubt without. In *CIKM 2012*, pages 1015–1024.
[9] N. Mishra, R. Saha Roy, N. Ganguly, S. Laxman, and M. Choudhury. Unsupervised query segmentation using only query logs. In *WWW 2011*, pages 91–92.
[10] S. Petrov, D. Das, and R. McDonald. A universal part-of-speech tagset. In *LREC 2012*, pages 2089–2096.
[11] R. Saha Roy, N. Ganguly, M. Choudhury, and S. Laxman. An IR-based Evaluation Framework for Web Search Query Segmentation. In *SIGIR 2012*, pages 881–890.
[12] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and wikipedia. In *WWW 2008*, pages 347–356.
[13] T. Tao and C. Zhai. An exploration of proximity measures in information retrieval. In *SIGIR 2007*, pages 295–302.
[14] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL 2003*, pages 173–180.