Random Walk Term Weighting for Information Retrieval

Roi Blanco
IRLab, Computer Science Department
University of A Coruña
Spain
rblanco@udc.es

Christina Lioma
Department of Computing Science
University of Glasgow
United Kingdom
xristina@dcs.gla.ac.uk

ABSTRACT

We present a way of estimating term weights for Information Retrieval (IR), using term co-occurrence as a measure of dependency between terms. We use the random walk graph-based ranking algorithm on a graph that encodes terms and co-occurrence dependencies in text, from which we derive term weights that represent a quantification of how a term contributes to its context. Evaluation on two TREC collections and 350 topics shows that the random walk-based term weights perform at least comparably to the traditional tf·idf term weighting, while they outperform it when the distance between co-occurring terms is between 6 and 30 terms.

Categories and Subject Descriptors: H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval

General Terms: Performance, Experimentation

Keywords: Random Walk Algorithm, TextRank

1. INTRODUCTION

Graph-based ranking algorithms like PageRank [4] have been used in citation analysis, social networks, and the analysis of the link structure of the Web. Generally, a graph-based ranking algorithm is a way of deciding on the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information. Recently, graph-based ranking algorithms were used with graphs extracted from text (TextRank [3]) and successfully applied to summarisation [1], text classification [2], keyphrase extraction [3], and word sense disambiguation [3].

We use the random walk graph-based ranking algorithm and its TextRank adaption [3] to derive term weights from textual graphs, similarly to [2]. Textual graphs encode term dependencies in text, which are defined as terms occurring within a maximum set distance of each other. Unlike previous studies [2, 3], we vary the window of co-occurring terms from 2 up to 40, and we apply the resulting random walk weights to Information Retrieval (IR). We hypothesise that the random walk term weights (rw) can perform at least comparably to traditional term frequency (tf) weights used in IR. We plug the random walk weights into the tf·idf weighting model, in the place of traditional term frequency weights. We evaluate the resulting rw·idf weighting using tf·idf as a baseline, on WT10G and Disks 4&5. We use tf·idf with pivoted document length normalisation, instead of term frequency normalisation, because it allows us to plug rw in the place of tf without further parameter tuning. Experiments show that random walk weights perform at least comparably to term frequency weights, while they outperform the latter when the distance between co-occurring terms is between 6 and 30.

SIGIR'07, July 23–27, 2007, Amsterdam, The Netherlands.
ACM 978-1-59593-597-7/07/0007.

2. RANDOM WALK TERM WEIGHTS

Graph-based ranking algorithms are a way of deciding the importance of a vertex within a graph, based on global information recursively drawn from the entire graph. When one vertex links to another one, it casts a vote for that other vertex. The higher the number of votes that are cast for a vertex, the higher the importance of the vertex. Also, the importance of the vertex casting the vote determines how important the vote itself is, and this information is taken into account by the ranking model. Hence, the score of a vertex is determined based on the votes that are cast for it and the score of the vertices casting these votes. Let $G = (V, E)$ be a directed graph with the set of vertices V and set of edges E, where E is a subset of $V \times V$. For a vertex V_i, let $In(V_i)$ be the set of vertices that point to it (predecessors), and let $Out(V_i)$ be the set of vertices that vertex V_i points to (successors). The score of a vertex V_i is defined as follows [4]:

$$S(V_i) = (1-d) + d \cdot \sum_{j \in In(V_i)} \frac{S(V_j)}{\mid Out(V_j) \mid} \qquad (1)$$

where d is a damping factor that can be set between 0 and 1, which has the role of integrating into the model the probability of jumping from a given vertex to another random vertex in the graph. In the context of the Web, this graph-based ranking algorithm implements a random walk model, where a user clicks on links at random with a probability d, and jumps to a completely new page with probability $1 - d$.

In order to build a graph encoding term dependencies from a document, we follow the steps used in the TextRank keyword extraction application [2]: 1) Add the document terms

		WT10G		Disks 4&5	
w	N	MAP	P@10	MAP	P@10
tf	-	0.1872	0.2969	0.2029	0.3912
rw	2	0.1855 (-0.9%)	0.3133 (+5.5%)	0.2102 (+3.6%)**	0.3871 (-1.0%)**
rw	4	0.1904 (+1.7%)*	0.3255 (+9.6%)*	0.2083 (+2.7%)**	0.3827 (-2.2%)**
rw	6	0.1921 (+2.6%)**	0.3408 (+14.8%)**	0.2210 (+8.9%)**	0.4028 (+3.0%)**
rw	8	**0.1972** (+5.3%)**	0.3429 (+15.5%)**	0.2247 (+10.7%)**	0.4104 (+4.9%)**
rw	10	0.1966 (+5.0%)**	**0.3449** (+16.2%)**	0.2277 (+12.2%)**	0.4153 (+6.2%)**
rw	15	0.1963 (+4.9%)**	**0.3449** (+16.2%)**	0.2330 (+14.8%)**	0.4269 (+9.1%)**
rw	20	0.1939 (+3.6%)**	0.3327 (+12.1%)**	0.2366 (+16.6%)**	0.4297 (+9.8%)**
rw	25	0.1935 (+3.4%)**	0.3296 (+11.0%)**	0.2394 (+18.0%)**	**0.4337** (+10.9%)**
rw	30	0.1891 (+1.0%)**	0.3265 (+10.0%)**	**0.2400** (+18.3%)**	0.4301 (+9.9%)**
rw	35	0.1860 (-0.6%)	0.3235 (+9.0%)	0.2383 (+17.4%)**	0.4209 (+7.6%)**
rw	40	0.1826 (-2.5%)	0.3112 (+4.8%)	0.2337 (+15.2%)**	0.4024 (+2.9%)**

		WT10G		Disks 4&5	
w	N	MAP	P@10	MAP	P@10
tf	-	0.2133	0.3540	0.2260	0.4169
rw	2	0.2197 (+3.0%)	0.3580 (+1.1%)	0.2376 (+5.1%)**	0.4273 (+2.5%)**
rw	4	0.2285 (+7.1%)**	0.3640 (+2.8%)**	0.2359 (+4.4%)**	0.4301 (+3.2%)**
rw	6	0.2268 (+6.3%)**	0.3650 (+3.1%)**	0.2472 (+9.4%)**	0.4498 (+7.9%)**
rw	8	0.2308 (+8.2%)**	0.3660 (+3.4%)**	0.2501 (+10.7%)**	0.4522 (+8.5%)**
rw	10	0.2322 (+8.9%)**	0.3740 (+5.6%)**	0.2525 (+11.7%)**	0.4546 (+9.0%)**
rw	15	**0.2328** (+9.1%)**	**0.3790**(+7.1%)**	0.2566 (+13.5%)**	0.4602 (+10.4%)**
rw	20	0.2266 (+6.2%)**	0.3730 (+5.4%)**	0.2604 (+15.2%)**	0.4558 (+9.3%)**
rw	25	0.2229 (+4.5%)**	0.3720 (+5.1%)**	0.2634 (+16.5%)**	0.4590 (+10.1%)**
rw	30	0.2219 (+4.0%)**	0.3720 (+5.1%)**	0.2647 (+17.1%)**	**0.4622** (+10.9%)**
rw	35	0.2207 (+3.5%)**	0.3710 (+4.8%)**	**0.2652** (+17.3%)**	0.4614 (+10.7%)**
rw	40	0.1826 (-14.4%)*	0.3112 (-12.1%)*	0.2624 (+16.1%)**	0.4534 (+8.8%)**

Table 1: MAP and P@10 of short (left) and long (right) queries. w is the weight used (term frequency (tf) or random walk (rw)). N is the window size of co-occurring terms. tf is the baseline. * (**) = statistical significance at $p < 0.05$ ($p < 0.01$) using the Wilcoxon matched-pairs signed-ranks test. Best scores are bold.

as vertices in a graph. 2) Identify terms co-occurring within a window of maximum size N. The window size is the maximum vicinity of a term, within which terms are considered co-occurrent. This is represented in the graph by a set of edges that connect this term to all the other terms in the window. 3) Iterate the graph-based algorithm until convergence. 4) Sort vertices based on their final score. After the graph is constructed and the edges are in place, we apply the TextRank algorithm. We set the maximum number of iterations to 100, the damping factor to 0.85, the convergence threshold to 0.0001, and the initial weight of each graph node to 0.25, following [2, 3]. We set window size N to 2, 4, 6, 8, 10, 15, 20, 25, 30, 35, and 40.

3. EVALUATION

To evaluate the random walk-based term weights, we plug them in place of term frequency weights in tf·idf, thus producing rw·idf. We evaluate retrieval performance using rw·idf, with tf·idf as a baseline, on WT10G (topics 451-550) and Disks 4&5 without the Congressional Record (topics 301-450 & 601-700)[1]. We experiment with short (title-only) and long (title-description) queries. We apply stopword removal and Porter stemming, and use the Terrier IR platform[2].

Table 1 contains the Mean Average Precision (MAP) and Precision at 10 (P@10) scores, for short and long queries. Overall, performance improves when random walk weights are used instead of term frequency weights, for terms co-occurring within a window of between to 6 and 30 terms, at all times, and in a similar range for both query lengths tested. Most of these improvements are very statistically significant ($p < 0.01$, Wilcoxon matched-pairs signed-ranks test). Varying the window size of co-occurring terms affects performance, with best scores noted between N=8 (+5.3% MAP, short queries, WT10G) and N=35 (+17.3% MAP, long queries, Disks 4&5). Best window sizes for MAP are close or even identical to best window sizes for P@10. This indicates that term dependency is modelled accurately within those window sizes. Comparing the two collections, we see that MAP improves more for Disks 4&5, (+18.3% (+17.3%), short (long) queries), than for WT10G, (+5.3% (+9.1%), short (long) queries), with random weights. Also, the best MAP window sizes are smaller for WT10G, (N=8 (15), short (long) queries), than for Disks 4&5 (N=30(35), short (long) queries). Both observations (smaller MAP improvement and smaller best MAP windows for WT10G) may be because WT10G is a Web collection that contains more noise, than Disks 4&5. Hence, modelling term dependency appears more effective when the co-occurring terms contain less noise. Finally, we report that, for most terms, tf and rw are positively correlated (Pearson's correlation coefficient >0.7). This correlation decreases as window size increases, since including more co-occurring terms alters rw, and co-occurring terms tend to become less relevant as window size increases.

[1]TREC datasets: http://trec.nist.gov/
[2]http://ir.dcs.gla.ac.uk/terrier/

4. CONCLUSIONS

We used a graph-based ranking model for text processing, namely the TextRank random walk algorithm, to derive term weights for Information Retrieval. Evaluation on two TREC collections and 350 topics showed that the random walk weights, when plugged into the tf·idf weighting scheme, perform at least comparably to tf·idf, while they significantly outperform it when the distance between co-occurring terms is between 6 and 30 terms. These results, which are consistent for two collections and queries of two different lengths, agree with previous findings that random walk weights can be successfully used in automatic text processing [1, 2, 3]. To our knowledge, this is the first time random walk weights are used as term weights in Ad-hoc retrieval. Future work includes integrating random walk weights into the weighting model in more refined ways, for example as probabilities, and in the context of Language Modelling.

Acknowledgements: Author 1 was supported by SEUI and FEDER funds under project MEC TIN2005-08521-C02 and "Xunta de Galicia" under project PGIDIT06PXIC10501PN.

5. REFERENCES

[1] G. Erkan and D. Radev. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. In *Journal of Artificial Intelligence Research*. 22, 457-479, 2004.

[2] S. Hassan and C. Banea. Random-Walk Term Weighting for Improved Text Classification. In *Proceedings of TextGraphs: 2nd Workshop on Graph Based Methods for Natural Language Processing*. ACL, 53-60, 2006.

[3] R. Mihalcea and P. Tarau. TextRank: Bringing Order into Texts. In *Proceedings of Empirical Methods in Natural Language Processing*. ACL, 404-411, 2006.

[4] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998.