Wikilinking and Adhocsearching

Dylan Jenkinson, Andrew Trotman, Kai-Cheung Leung

Department of Computer Science

University of Otago

Dunedin

New Zealand

{dylan, andrew, kcleung}@cs.otago.ac.nz

**Abstract.** The University of Otago submitted six runs to the Link-the-Wiki track with the top run placing nth. Three element runs and three passage runs were submitted to the Relevant in Context task of the ad hoc track. The best Otago run was a whole-document run placing 7th. The best Otago passage run placed 13th while the best Otago element run placed 31st. There were a total of 40 runs submitted to the task. This result reinforced our prior belief that passages are better answers than elements and that the most important aspect of the focused retrieval is the identification of relevant documents..

Introduction

Otago participated in the Link-the-Wiki task in 2007 and produced runs that performed adequately however the results of Itakura & Clarke (Itakura and Clarke 2007) and of Geva (Geva 2007) were more successful at producing outgoing links. For 2008 Otago concentrated on reproducing and extending the work Itakura by including some of the findings of Geva, and including multiple targets per link.

To do this we first re-implemented their algorithm. This involved finding all phrases in the collection that were used as a link to another document, and recording this. We then found the most linked to document for each phrase, and stored that, along with a value representing the strength of the link. When a topic was processed, the 250 strongest links were used.

One problem with the results from this implementation was the occurrence of duplicate links between the orphan document and other documents in the collection. Such duplicates, where they pointed from the same anchor point in the orphan to the same anchor point in the target document, were considered as errors, and would have pulled the MAP down as a result. We removed these duplicates from our results. We also looked at weather paying attention to case would affect the results.

For incoming links, we tried several approaches. These included sorting the results based on the BM25 result from our search engine, learning weights for different terms in a search used to find documents related to the topic, and using the title of the topic for the search.

Stuff about how we did goes here.

Link Detection in the Wikipedia

The Link-the-Wiki task, first included in INEX in 2007, requires participants to automatically identify hypertext links between documents in the Wikipedia. The user model is that of a user who creates a new Wikipedia entry and would like to link that entry to pre-existing entries in the Wikipedia (and *vice versa*).

The production of a new article can be simulated by taking an existing Wikipedia document and removing all trace of it from the collection. Link identification software can then be run over the collection and the orphaned document. A comparison of the automatically generated links to the original collection gives some measure of the quality of the link detection system – that is, the original links are considered to be the gold-standard by which systems are compared.

Exactly this approach was taken in the INEX 2007 Link-the-Wiki track, and was used again for document-to-document linking in 2008. In 2008, 6600 documents (about 1% of the document collection) were randomly selected and orphaned for whole document link detection.

New in 2008 is the anchor-to-BEP linking task, in which the task is to identify the best orphan anchor from which to link from and the best-entry-point in the target document from which to link to. Unlike document-to-document linking, anchor-to-BEP linking requires manual assessment because the Wikipedia documents are not a priori marked-up in this way. For 2008, 50 anchor-to-BEP documents suggested by task participants and were orphaned for the experiment. A limit of 50 anchors per document was imposed (for practical reasons) and at most each anchor could link to 5 locations in the Wikipedia.

We examine the problem of link identification by first examining outgoing links (from the orphan to the collection) then incoming links (from the collection to the document).

As shown in (Jenkinson and Trotman 2007)and (Kamps, Koolen et al. 2007), the best entry point of a document in the Wikipedia collection is considered to be the start of the document. Therefore, for the second set of results for Link the Wiki, the links all pointed to the start of the documents.

Detecting Outgoing Links

Although the Otago runs in 2007 were adequate, those of Itakura and Clarke (Itakura and Clarke 2007) were substantially better – we chose, therefore, to investigate methods of improving their technique. It should be noted that the Itakura & Clarke algorithm relies on a pre-existing heavily interlinked document collection (such as the Wikipedia). In the case where no prior links exist in the collection the techniques of Geva (Geva 2007) which were also successful in INEX 2007 can be used.

The Itakura & Clarke Algorithm

The Itakura & Clarke algorithm relies entirely on pre-existing links between documents within the document collection. Of the link types available in the collection, only the <collectionlink> type is utilized because the other link types do not link between two documents in the collection (for example, a <wikipedialink> links from a document in the collection to a document in the Wikipedia that is not in the INEX collection.

Initially a list of all the links within the document collection is created. This is generated by parsing each document in the collection and extracting the anchor text of the link and the target document id.

Next and from the output of the previous stage, a list of target documents is created for each unique anchor text in the collection. For a given anchor text in the collection, the most frequent target is most likely to be the correct target.

For each anchor text / target pair a strength value ( is constructed

*=np/af*

where *np* is the number of documents that link from the anchor to the target and *af* is the number of pages in which the anchor text occurs.

An orphaned document is then parsed and the first location of each anchor in the pre-generated list is located. For overlapping anchors (for example, “Lennon” and “John Lennon”) the longest possible anchor is chosen as a longer anchor is more likely to be correct than a short anchor. A limit of 250 anchors per document was enforced by the Link-the-Wiki track definition.

Small Improvements

After implementing the Itakura & Clarke algorithm verbatim we identified a number of small improvements.

The algorithm defines the anchor text as all text occurring between the tags regardless of case and punctuation. Anchor texts often contain punctuation at the end thus creating a distinction between “John Lennon” and “John Lennon.”. We stripped punctuation from the anchors.

Anchor texts beginning at the start of a sentence are capitalized for grammatical reasons so the algorithm converts the text into lower case. Unfortunately this results in a distinction between “unfinished music” and “Unfinished Music” (the two part experimental work by John Lennon). Geva [[CITE]] identifies the important of case in link detection so we dropped the case conversion step.

Finally, we over-weighted  for terms containing capitalization by adding a constant, *p*. We expect those terms to be proper nouns and thus Wikipedia entries. Experiments with values between 0 and 1.0 suggested that NXX was best.

**Fig. 1.** **Small improvements on the Itakura & Clarke algoritm (labeled Waterloo) are seen when the anchor list is deduplicated by removing punctuation (Deduplicated), when case folding is removed (Upper Case) and when uppercase anchors are preferred over lowercase anchors (Weighted Upper Case).**

Figure 1 compares our implementation of the algorithm to the original on the INEX 2007 Link-the-Wiki topics. The line labeled “Waterloo” is the Itakura & Clarke run as submitted. Removing punctuation (deduplicated) from the anchor list improves the algorithm, removing case folding (Upper Case) leads to further improvements. Our best run “Weighed Upper Case” included dedplication, removed case folding, and weighted  with NXX.

Figure . Comparison of Different Methods for Generating Outgoing Links

Incoming Links

Multi Search Methods

At INEX 2007, we detected incoming links by taking the results we got for outgoing links and switching them around. This lead to adequate result, with our best run performing best at early precision (0.751 P@5), but left much room for improvement. In 2008, we looked at a few different approaches to improving our results.

The first approach took the work that we did last year [[Our paper last year]] and extended it some. Last year, our approach to finding appropriate documents went like this. First we striped the orphan document of its XML markup. Then we found the number of occurrences of each word, ignoring case, in the orphan. Ignoring stop words, we generated a value for each word, showing how over represented that word was in the document, compared to the entire collection. We then selected a number of words that scored the highest on over representation and preformed a series of searches to find documents related to the orphan. These searches each produced a batch of results. The top *n* results from each search preformed for the orphan were then used as results.

One thing that was not done with these results was to mix the results from separate searches for the same run together. Instead the results from each search were simply placed down, in their entirety, before those of subsequent searches. If three searches were done, then the results from the first search would go entirely before the results of the second search, and the second search entirely before the third search, even though there might be better results at the top of the third search then there were at the bottom of the second search. To extend our results from last year we sorted the results that we produced into a single merged set, based on the BM25 ranking that came out of the search engine. We took the results from all the searches that were preformed for a run and sorted them together. This would have pushed those with a good BM25 score to the top, and those with a bad BM25 score to the bottom, irrespective of which search they came from. We preformed the same process again, but this time weighted the results based on what search they came from. The weight for this was generated by adding the over representation scores for each term in the search, and then multiplying the score from the search engine based on this. These results were then sorted together based off of their weighted BM25 score, pushing those with good scores to the top, and those with bad scores to the bottom, again irrespective of the search that they came from.

In the 2007 results, our best run was with four terms per search, and two total searches. Figure 4 shows us that as the number of search terms grew beyond four, the performance decreased. It should be noted that for all searches with five or more terms, there was only one search preformed by the system, as these individual searches would produce the allowed 250 results. When trying these two sorting methods, the second, weighted search, approach produced the best score, with two terms per search, and three searches.

Figure 2 shows us the top preformer from each of the different merge methods, as well as our best result from 2007. Both last years best run (*2007 Best*) and the simple unweighted sort (*Unweighted Sort*) score better at early precision then the weighted sort (*Weighted Sort*) does, but weighted sort gets the better MAP, helped by better precision at later recall.

Table . MAP scores for different runs

|  |  |
| --- | --- |
| **Run** | **MAP** |
| Top Official Run from 2007 | 0.484 |
| Best Weighted Sort Run | 0.35 |
| Top Otago Run from 2007 | 0.339 |
| Best Simple Sort Run | 0.319 |

Figure . Comparison of top performing runs from the different merging methods and 2007 best run

Single Search Methods

Up to this point, all results had been generated using a series of searches returning part of the result set. How would this compare to the same process using just a single search that returned a partial or full set of results?

To start with, we preformed the same series of steps as the multi search experiments, but with only a single search being preformed, and only returning the results that the first search of the multi search system would have. This was preformed to compare how the first search was contributing to the overall results, and how the extra searches, and results added from them, were affecting the results.

The next idea we looked at was how a single search, returning the whole 250 allowed results, would compare to the mixed results returned by the multi search methods, and the single partial result set.

As can be seen in Figure 3 below, the single search that returned only the first part of the results (*Single Partial Results*) made up a large portion of the overall results returned by the multiple search method (*Multiple Searches*). The single search that returned all allowed results (*Single Full Results*) preformed better then the multiple search method. This suggested that methods using just a single search and returning the full number of allowed results would perform better than our previous attempts.

Figure . Single searches with partial and full results compared to multiple search method.

Learning Weights for Individual Terms in the Search

At this point, when searches were being preformed, all terms were being considered with equal value, irrespective of their original over representation scores. But terms in the document had different over representation scores, showing that they had different importance to the meaning of the document. This suggested that different terms in a search should have different influence on the results of the search.

To allow this difference in influence, weighting on individual terms was introduced. A single search was preformed, for each search length from 2 to 10 terms, and each term within the search was given its own weight. These weights were passed through to BM25 where a weighted term frequency (*ctf*) was used instead of a normal term frequency. The *ctf*, from [[your paper]] and [[BM25F paper]], is the frequency of the term (*tf*) in a field (*p*) in the document, weighted by the field weight (*Cp*). In our case, we used the whole document for *p*, which makes the *ptf* the same as the normal document *tf*. *Cp* is the passed in weight for that particular term.

|  |  |
| --- | --- |
| *ctf* = *ptf* · *Cp* | (**1**) |

We used a genetic algorithm (GA) to breed the weights. Each run of the GA used population sizes of 50, with a total of 10 generations. Crossover was 0.9, mutation 0.05, and reproduction 0.05. We used elitist breeding. Several runs were done for each of the search lengths. The weights generated were between 0.0 and 1.0, inclusive. Figure 4 shows the MAP scores of the best set of weights. Table 2 shows the values of the weights that make up the best set for each number of terms used in the searches.

Table . Best set of weights for different numbers of terms in a search

|  |  |
| --- | --- |
| **Number of Search Terms** | **Set of Weights** |
| 2 | 0.96, 0.95 |
| 3 | 0.99, 0.96, 0.04 |
| 4 | 0.97, 0.73, 0.05, 0.06 |
| 5 | 0.95, 0.83, 0.14, 0.1, 0.01 |
| 6 | 0.89, 0.97, 0.44, 0.41, 0, 0.06 |
| 7 | 0.8, 0.95, 0.75, 0.29, 0, 0.07, 0.25 |
| 8 | 1, 0.88, 0.14, 0.05, 0, 0.22, 0.08, 0.19 |
| 9 | 0.87, 0.81, 0.36, 0.26, 0, 0.22, 0.29, 0.2, 0.01 |
| 10 | 0.9, 0.99, 0.77, 0.55, 0.35, 0.08, 0.19, 0.16, 0, 0.19 |

Figure . MAP scores generated from best set of learned weights for number of search terms 2-10, as well as searches using an increasing number of .

Using Parts of the Document for the Search

The title of a Wikipedia document tells the user what the document is about. What kind of result would we get if we used just the title of a document to find appropriate documents for incoming links? How would this compare to using the first paragraph of the document, or the whole document itself?

The title of a Wikipedia document is the name of the subject that the document covers. These titles are often nouns, such as the name of a town (such as “Manhattan”), or the name of a vehicle (such as “Chevrolet Corvette”). On those occasions where this does not hold, the page is being used for some other purpose, such as disambiguation between different pages with the same name. Therefore, it seems reasonable to believe that the name of the document will contain enough information to find other pages on the same topic.

Each Wikipedia document has a title, which sits between the <name> tag. To produce our results for this experiment, we generated a list of all the titles of documents and the corresponding document number. For each topic, we looked up the title in this list, and then did a search using the title, removing duplicates and stop words. This approach could hold problems, in situations where the title is made up entirely of stop words, such as “The The”, but no attempt was made to deal with this issue at this point.

Many of the documents in the Wikipedia have a first section which contains an overview of the entire documents contents. These sections appear to be designed to give a user the ability to quickly garner the necessary information to decide if the document will help them with their information needs. Based on this observation, we decided to compare the content of such sections with the results of the title only.

To select these first sections from the Wikipedia collection documents, a program was written that looked for the first <title> tag in the document. <title> tags appear to be used at the start of sections of the document to describe what that sections are about. We used these tags to define the end of the first section of the document, as there was no discernable structure used to define those sections we wanted, but there was often a <title> tag defining the first non-overview section of the document. For those documents that lacked any <title> tag, we used the whole document instead.

As a comparison measure, we choose to use the whole document for the search as well. This allowed us to see how the other document parts methods compared against all the possible information in the document.

Figure . Different parts of the document used as a single search

As can be seen in Figure 5 the title by itself (*Title Only*) out performs the other parts of the document substantially. The first paragraph (*First Paragraph*) also performs well, while the document as a whole (*Whole Document*) is the lowest scoring of the set.

Conclusions

Some conclusions go here.

Acknowledgements

Funded in part by a University of Otago Research Grant.

References

Geva, S. (2007). GPX: Ad-Hoc Queries and Automated Link Discovery in the Wikipedia. Focused Access to XML Documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007 Dagstuhl Castle, Germany, December 17-19, 2007. Selected Papers, Springer-Verlag: 404-416.

Itakura, K. Y. and C. L. Clarke (2007). University of Waterloo at INEX2007: Adhoc and Link-the-Wiki Tracks. Focused Access to XML Documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007 Dagstuhl Castle, Germany, December 17-19, 2007. Selected Papers, Springer-Verlag: 417-425.

Jenkinson, D. and A. Trotman (2007). Wikipedia Ad Hoc Passage Retrieval and Wikipedia Document Linking. Focused Access to XML Documents: 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2007 Dagstuhl Castle, Germany, December 17-19, 2007. Selected Papers, Springer-Verlag: 426-439.

Kamps, J., M. Koolen, et al. (2007). Where to Start Reading a Textual XML Document? 30th ACM SIGIR Conference on Information Retrieval, Amsterdam, The Netherlands.