INEX 2006 Evaluation Measures

Mounia Lalmas[1], Gabriella Kazai[2], Jaap Kamps[3], Jovan Pehcevski[4], Benjamin Piwowarski[5], and Stephen Robertson[2]

[1] Queen Mary, University of London
[2] Microsoft Research Cambridge
[3] University of Amsterdam
[4] AxIS Project group, INRIA Rocquencourt, France
[5] Yahoo! Research

Abstract. This paper describes the official measures of retrieval effectiveness employed at the ad-hoc track of INEX 2006.

1 Introduction

Since its launch in 2002, INEX has been challenged by the issue of how to measure an XML retrieval system's effectiveness. The main complication comes from the necessity to consider the dependency between elements when evaluating effectiveness. Unlike traditional IR, users in XML retrieval have access to other, structurally related elements from returned result elements. They may hence locate additional relevant information by browsing or scrolling. This motivates the need to consider so-called near-misses, which are elements from where users can access relevant content, within the evaluation. The alternative, to ignore near-misses, would lead to a strict evaluation scenario, especially when dealing with fine-grained XML documents.

As discussed in Section 2, the ad hoc track at INEX 2006 has four retrieval tasks, namely focused task, thorough task, relevant in context task, and best in context task. INEX 2006 uses three sets of metrics to evaluate these tasks:

- The XCG metrics introduced at INEX 2005 [3] are used to evaluate the thorough and the focused retrieval tasks (Section 4 and 5, respectively).
- The HiXEval metrics originally proposed in [7] were adapted to evaluate the relevant in context retrieval task (Section 6).
- The BPRUM metrics originally defined in [8] were adapted to evaluate the best in context retrieval task. A set-based measure was also defined to evaluate this task (Section 7).

This paper is organized as follows. In 2, we describe the INEX 2006 ad hoc retrieval tasks, including their motivations. In Section 3, we describe how relevance is defined in INEX 2006. The evaluations of each task are described in the next four subsequent sections (Sections 4 to 7). We finish the paper with conclusions and our plans for INEX 2007.

2 Ad-hoc retrieval tasks

The main INEX activity is the ad-hoc retrieval task, where the collection consists of XML documents (Wikipedia articles), composed of different granularity of nested XML elements, each of which represents a possible unit of retrieval. A major departure from traditional IR is that XML retrieval systems need not only score elements with respect to their relevance to a query, but also determine the appropriate level of element granularity to return to users, thus allowing focussed access to XML documents. The user's query may also contain structural constraints or hints in addition to the content conditions. In addition, the output of an XML retrieval system may follow the traditional ranked list presentation, or may extend to non-linear forms, such as grouping of elements per document.

Up to 2004, ad-hoc retrieval was defined as the general task of returning, instead of whole documents, those XML elements that are most relevant to the user's query. In other words, systems should return components that contain as much relevant information and as little irrelevant information as possible. Within this general task, several sub-tasks were defined, where the main difference was the treatment of the structural constraints.

However, within this general task, the actual relationship between retrieved elements was not considered, and many systems returned overlapping elements (e.g. nested elements). This had very strong implications with respect to measuring effectiveness, where approaches that attempted to implement a more focussed approach performed poorly. As a result, the focussed task was defined in 2005, intended for approaches concerned with the focussed retrieval of XML elements, i.e. aiming at targeting the appropriate level of granularity of relevant content that should be returned to the user for a given topic. The aim was for systems to find the most relevant element on a path within a given document containing relevant information and return to the user only this most appropriate unit of retrieval. Returning overlapping elements was not permitted. The INEX ad-hoc general task, as carried out by most systems up to 2004, was renamed in 2005 as the thorough sub-task.

Within all the focused and thorough tasks, the output of XML retrieval systems was assumed to be a ranked list of XML elements, ordered by their presumed relevance to the query. User studies [10] suggested that users were expecting to be returned elements grouped per document, and to have access to the overall context of an element. The fetch & browse task was introduced in 2005 for this reason. The aim was to first identify relevant documents (the fetching phase), and then to identify the most relevant elements within the fetched documents (the browsing phase).

In 2005, no explicit constraints were given regarding whether returning overlapping elements within a document was allowed. The rationale was that there should be a combination of how many documents to return, and within each document, how many relevant elements to return. In 2006, the same task, renamed the relevant in context task, required systems to return for each document an unranked set of non-overlapping elements, covering the relevant material in the document. In addition, a new task was introduced in 2006, the best in context

task, where the aim was to find the best-entry-point, here a single element, for starting to read documents with relevant information. This new task can be viewed as the extreme case of the fetch & browse approach, where only one element is returned per document.

To summarize, INEX 2006 investigated the following four ad-hoc retrieval tasks defined as follows [1]:

- Thorough: This task asks systems to estimate the relevance of all XML elements in the searched collection and return a ranked list of the top 1500 elements.
- Focused: This task asks systems to return a ranked list of the most focused XML elements, where result elements should not overlap (e.g. a paragraph and its container section should not both be returned). Here systems are forced to choose from overlapping relevant elements those that represent the most appropriate unit of retrieval.
- Relevant in Context[6]: This task asks systems to return to the user the most focused, relevant XML elements clustered by the unit of the document that they are contained within. An alternative way to phrase the task is to return documents with the most focused, relevant elements indicated (e.g. highlighted) within.
- Best in Context: This task asks systems to return a single best entry point to the user per relevant document.

3 Relevance Assessments

In INEX 2006, relevance assessments were obtained by assessors highlighting relevant text fragments in the documents, which correspond wikipedia articles in 2006 (see the overview paper in this proceedings). XML elements that contained some highlighted text were then considered as relevant (to varying degree). A default assumption here is that if an XML element is relevant (to some degree), then its ascendant elements will all be relevant (to varying degrees) due to the subsumption of the descendant elements' content. For each relevant XML element, the size of the contained highlighted text fragment (in number of characters) is recorded as well as the total size of the element (again, in number of characters). These two statistics form the basis of calculating an XML element's relevance score, which in 2006 corresponds to its specificity score.

The specificity score, $spec(c_i) \in [0, 1]$ of a component c_i is calculated as the ratio of the number of highlighted characters contained within the XML element $hl(c_i)$ to the length of the element $len(c_i)$.

$$spec(c_i) := \frac{hl(c_i)}{len(c_i)} \tag{1}$$

[6] The run submission DTD refers to this task as "AllInContext".

4 Evaluation of the Thorough task

4.1 Assumptions

This task is based on the assumption that all XML elements of a searched collection can be ranked by their relevance to a given user query. The task of a system here is then to return a ranked list of the top 1500 relevant XML elements, in descreasing order of relevance. The goal of this task is to test a system's ability to produce the correct ranking. No assumptions are made regarding the presentation of the results to the user: the output of a system here can simply be considered as an intermediate stage, which may then be processed for displaying to the user (e.g. filtered, clustered, etc.). Therefore, issues, like overlap (e.g. when a paragraph and its container section are both returned) are ignored during the evaluation of this task.

4.2 Evaluation measures

Two indicators of system performance were employed in the evaluation of the Thorough task: Effort-precision/gain-recall (ep/gr) graph and Mean Average effort-precision ($MAep$). These are both members of the eXtended Cumulated Gain (XCG) measures [3], which were chosen as they are extensions of the Cumulated Gain based metrics of [2]. These were developed specifically for graded (non-binary) relevance values and with the aim to allow IR systems to be credited according to the retrieved documents' degree of relevance.

From the family of XCG measures, ep/gr and $MAep$ were selected as they provide an overall picture of retrieval effectiveness across the complete range of recall. The motivation for this choice is the recall-oriented nature of the task, e.g. rank all elements of the collection and return the top 1500 results. $MAep$ summarises retrieval effectiveness into a single number, while an ep/gr graph allows for a more detailed view, ploting ep at 100 recall points $0.01, 0.02, ..., 1$.

These measures are implemented within the XCGEval package of the EvalJ java project - please refer to Appendix 9.

Gain value. The definition of all XCG measures is based on the underlying concept of the value of gain, $xG[i]$, that a user obtains when examining the i-th result in the ranked output of an XML IR system. Given a ranked list of document components, where the XML element IDs are replaced with their relevance scores, the cumulated gain at rank i, denoted as $xCG[i]$, is computed as the sum of the relevance scores up to that rank:

$$xCG[i] := \sum_{j=1}^{i} xG[j] \tag{2}$$

Assuming that users prefer to be returned more relevant elements first, an ideal gain vector, xI, can be derived for each query by filling the rank positions

with the relevance scores of the relevant elements in decreasing order of their relevance scores. The corresponding cumulated ideal gain vector is denoted as xCI and is calculated analogue to $xCG[i]$.

Effort-precision/gain-recall. Effort-precision at a given cumulated gain value, r, measures the amount of relative effort (where effort is measured in terms of number of visited ranks) that the user is required to spend when scanning a system's result ranking compared to the effort an ideal ranking would take in order to reach the given level of gain (illustrated by the horizontal line drawn at the cumulated gain value of r in Figure 1):

$$ep[r] := \frac{i_{ideal}}{i_{run}} \tag{3}$$

where i_{ideal} is the rank position at which the cumulated gain of r is reached by the ideal curve and i_{run} is the rank position at which the cumulated gain of r is reached by the system run.

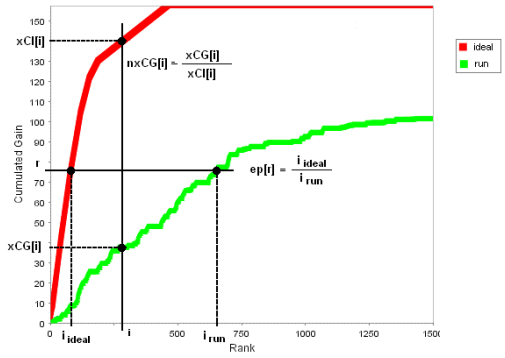

Fig. 1. Calculation of $nxCG$ and effort-precision (ep)

By scaling the recall axis to $[0, 1]$ (i.e. dividing by the total gain), effort-precision can be measured at arbitrary recall points, $gr[i]$ [4]:

$$gr[i] := \frac{xCG[i]}{xCI[n]} = \frac{\sum_{j=1}^{i} xG[j]}{\sum_{j=1}^{n} xI[j]} \tag{4}$$

where n is the total number of relevant elements.

As with standard precision/recall, for averaging across queries, interpolation techniques are necessary to estimate effort-precision values at non-natural gain-recall points, e.g. at standard recall points $0.1, .., 1$.

The non-interpolated mean average effort-precision, denoted as $MAep$, is calculated by averaging the effort-precision values obtained for each rank where

a relevant document is returned[7]. For not retrieved relevant elements a precision score of 0 is used.

4.3 Evaluation parameters

For the evaluation of the Thorough task, the XCG measures require four main parameters: 1) the gain value of the i-th element in a system's ranking, $xG[i]$, 2) the range for i, 3) the gain value of the j-th element in the ideal ranking, $xI[j]$, and 4) the range for j.

For the Thorough task, both $xG[i]$ and $xI[j]$ are calculated using the element's specificity value:

$$xG[i] = spec(c_i) \tag{5}$$

$$xI[j] = spec(c_j) \tag{6}$$

where c_i and c_j denote elements, the specificity score is given in Equation 1.

The range for i is $[0, 1500]$, where 1500 is the maximum length of a result list that participants could submit. The range for j is $[0, n]$, where n is the total number of relevant XML elements for the given query.

5 Evaluation of the Focused task

5.1 Assumptions

In this task, systems are asked to return the ranked list of the top 1500 "most focused" XML elements that satisfy an information need, without returning overlapping elements. This task is motivated by user study findings [10], which show that users get frustrated when overlapping results are presented to them in a ranked list. The task is similar to the Thorough task in that it requires a ranking of XML elements, but here systems are required not only to estimate the relevance of elements, but also to decide which element(s), from a tree of relevant nodes, are the most focused non-overlapping result(s).

5.2 Evaluation measures

The normalised cumulated gain $nxCG[RCV]$ measure, from the XCG family of measures, was used in the evaluation of the Focused task. System performance was reported at several rank cutoff values (RCV). Low RCV values were used, reflecting that users are typically only expected to scan the top part of a result list.

The $nxCG$ measures are also implemented within the XCGEval package of the EvalJ java project - please refer to Appendix 9.

[7] Note that, unlike with precision/recall, it is necessary to use interpolation on the ideal curve to obtain $MAep$.

Normalised cumulated gain. For a given query, the normalised cumulated gain ($nxCG$) measure is obtained by dividing a retrieval run's xCG vector by the corresponding ideal xCI vector:

$$nxCG[i] := \frac{xCG[i]}{xCI[i]} \tag{7}$$

For a given rank i, the value of $nxCG[i]$ reflects the relative gain the user accumulated up to that rank, compared to the gain he/she could have attained if the system would have produced the optimum ranking. As illustrated in Figure 1, $nxCG$ is calculated by taking measurements on both the system and the ideal rankings' cumulated gain curves along the vertical line drawn at rank i. Here, rank position is used as the control variable and cumulated gain as the dependent variable.

5.3 Evaluation parameters

As with the Thorough task, the evaluation of the Focused task requires four main parameters: 1) the gain value of the i-th component in a system's ranking $xG[i]$, 2) the range for i, 3) the gain value of the j-th component in the ideal ranking $xI[j]$, and 4) the range for j.

First, we define two different recall bases. The full recall-base is the list of all components that contain any relevant matter (which therefore includes all parents of any such element), already used in the Thorough task. The ideal recall-base is a subset of the full recall-base, where overlap between relevant reference elements is removed so that the identified subset represents the set of ideal answers, i.e. the most focused elements that should be returned to the user. $xG[i]$ is based on the full recall-base; $xI[j]$ is based on the ideal recall-base (see below for its definition and the specification of the selection of the ideal recall-base). The actual gain values, are identical to that used for the Thorough task (Section 4.3, see Equations 5 and 6, respectively).

The selection of ideal elements into the ideal recall-base is done by traversing an article's XML tree and selecting from the set of overlapping relevant elements, those with the highest gain value. The methodology to traverse an XML tree and select the ideal elements is as follows [9]: Given any two elements on a relevant path[8], the element with the higher score is selected. In case two elements' scores are equal, the one higher in the tree is chosen (i.e. parent/ascendant). The procedure is applied recursively to all overlapping pairs of elements along a relevant path until one element remains. After all relevant paths in a document's tree have been processed, a final filtering is applied to eliminate any possible overlap among ideal elements, keeping from two overlapping ideal paths the shortest one.

The range for i is $[0, 1500]$, where 1500 is the maximum length of a result list that participants could submit. The range for j is $[0, n]$, where n is the total number of relevant XML elements in the ideal recall-base.

[8] A relevant path is a path in an article file's XML tree, whose root element is the article element and whose leaf element is a relevant element.

6 Evaluation of the Relevant in Context task

6.1 Assumptions

The Relevant in Context task is document (here Wikipedia article) with a twist, where not only the relevant articles should be retrieved but also a set of XML elements repre- senting the relevant information within each article. Phrased differ- ently, the system should return the relevant information (captured by a set of XML elements) within the context of the full article. The task corresponds to an end-user task where focused retrieval results are grouped per article, in their original document order, providing access through further navigational means. This assumes that users consider the article as the most natural unit of retrieval, and prefer an overview of relevance in their context. Interactive ex- periments at INEX provided support for this task [10]. Moreover, the task directly corresponds with the assessors task at INEX, where assessors are asked to highlight the relevant information in a pooled set of articles.

In this task, there is a fixed result presentation format defined. Systems are expected to return the user sets of most focused elements within a relevant XML Wikipedia article that the elements are contained within. The Wikipedia articles are to ranked in decreasing order of relevance. The assumption is that users would view the complete articles, where the most focused elements would appear highlighted. There is no ranking of the contained XML elements within a document (users may simply follow reading order).

The task is based on the INEX 2005 Fetch-And-Browse retrieval strategy [6]. The aim of the Relevant In Context task is to first identify relevant articles (the fetching phase), and then to identify the most focused, relevant elements within the fetched articles (the browsing phase). The output of the fetching phase is a ranked list of articles, ranked according to their relevance to the query. In the browsing phase, we have a set of elements that cover the relevant information in the article. Note that the `//article[1]` element itself need not be returned, but is implied by any result element from it that is included in the result list. The set of result elements should not contain overlapping elements.

6.2 Evaluation measures

The evaluation of this task is based on a ranked list of articles, where per article-rank we obtain a score reflecting how well the retrieved set of elements corresponds to the relevant information in the article.

Score per article-rank For a retrieved article, the text retrieved by the selected set of elements is compared to the text highlighted by the assessor [7]. We calculate the following:

- *Precision*, as the fraction of retrieved text (in bytes) that is highlighted;
- *Recall*, as the fraction of highlighted text (in bytes) that is retrieved; and

– *F-Score*, as the combination of Precision and Recall using their harmonic mean, resulting is a score in [0,1] per article.

More formally, let a_r be an article assigned to a rank r in a ranked list of articles, and let e be an element that belongs to the set of retrieved elements $\mathcal{E}_{a_r}$. Let $rsize(e)$ be the amount of highlighted (relevant) text contained by e (if there is no highlighted text in the element, $rsize(e) = 0$). Let $size(e)$ be the total number of characters (bytes) contained by e, and let $Trel(a_r)$ be the total amount of (highlighted) relevant text for the article a_r.

We measure the fraction of retrieved text that is highlighted for article a_r as:

$$\mathtt{P}(a_r) = \frac{\sum\limits_{e \in \mathcal{E}_{a_r}} rsize(e)}{\sum\limits_{e \in \mathcal{E}_{a_r}} size(e)}$$

The $\mathtt{P}(a_r)$ measure ensures that, to achieve a high precision value for the article a_r, the set of retrieved elements for that article needs to contain as little non-relevant information as possible.

We measure the fraction of highlighted text that is retrieved for article a_r as:

$$\mathtt{R}(a_r) = \frac{\sum\limits_{e \in \mathcal{E}_{a_r}} rsize(e)}{Trel(a_r)}$$

The $\mathtt{R}(a_r)$ measure ensures that, to achieve a high recall value for the article a_r, the set of retrieved elements for that article needs to contain as much relevant information as possible.

The final score per article is calculated by combining the two precision and recall scores in the standard F-score (the harmonic mean) as follows:

$$\mathtt{F}(a_r) = \frac{2 \cdot \mathtt{P}(a_r) \cdot \mathtt{R}(a_r)}{\mathtt{P}(a_r) + \mathtt{R}(a_r)}$$

The resulting F-score varies between 0 (article without relevance, or none of the relevance is retrieved) and 1 (all relevant text is retrieved and nothing more). For retrieved non-relevant articles, $\mathtt{P}(a_r) = \mathtt{R}(a_r) = \mathtt{F}(a_r) = 0$.

Scores for ranked list of articles We have a ranked list of articles, and for each article we have an F-score $\mathtt{F}(a_r) \in [0, 1]$. Hence, we need a generalized measure, and we utilise the most straightforward generalization of precision and recall as defined by Kekäläinen and Järvelin [5, p.1122-1123].

Over the ranked list of articles, we calculate the following:

– *generalized Precision* ($\mathtt{gP}[r]$), as the sum of F-scores up to an article-rank, divided by the article-rank; and
– *generalized Recall* ($\mathtt{gR}[r]$), as the number of articles with relevance retrieved up to an article-rank, divided by the total number of articles with relevance.

More formally, let us assume that for an INEX 2006 topic there are in total $Numrel$ articles with relevance, and let us also assume that the function $\texttt{rel}(a_r) = 1$ if article a_r contains relevant information, and $\texttt{rel}(a_r) = 0$ otherwise. At each rank r of the list of ranked articles, generalized Precision is defined as:

$$\texttt{gP}(r) = \frac{\sum_{i=1}^{r} \texttt{F}(a_i)}{r}$$

At each rank r of the list of ranked articles, generalized Recall is defined as:

$$\texttt{gR}(r) = \frac{\sum_{i=1}^{r} \texttt{rel}(a_i)}{Numrel}$$

These generalized measures are completely compatible with the standard precision/recall measures used in traditional information retrieval. Specifically, the Average generalized Precision (AgP) for an INEX 2006 topic can be calculated by averaging the generalized Precision at natural recall points where generalized Recall increases. That is, averaging the generalized Precision at ranks where an article with relevance is retrieved (the generalized Precision of non-retrieved articles with relevance is 0).

More formally, if $\mathcal{R}$ represents the ranked list of articles returned by an XML retrieval system, the Average generalized Precision is defined as:

$$\texttt{AgP} = \frac{\sum_{i=1}^{|\mathcal{R}|} \texttt{rel}(a_i) \cdot \texttt{gP}(i)}{Numrel} = \frac{\sum_{i=1}^{|\mathcal{R}|} \texttt{rel}(a_i) \cdot \texttt{gP}(i)}{\sum_{i=1}^{|\mathcal{R}|} \texttt{rel}(a_i)} \cdot \frac{\sum_{i=1}^{|\mathcal{R}|} \texttt{rel}(a_i)}{Numrel} = \frac{\sum_{i=1}^{|\mathcal{R}|} \texttt{rel}(a_i) \cdot \texttt{gP}(i)}{\sum_{i=1}^{|\mathcal{R}|} \texttt{rel}(a_i)} \cdot \texttt{gR}(|\mathcal{R}|)$$

When looking at a set of topics, the Mean Average generalized Precision (MAgP) is simply the mean of the average generalized Precision scores per topic.

6.3 Results reported at INEX 2006

For the AllinContext task we report the following measures over all topics:

- Mean Average Generalized Precision (MAgP)
- generalized Precision at early ranks (gP[5, 10, 25, 50])

The official Relevant in Context evaluation is based on the overall MAgP measure. All results are accessible on the INEX 2006 website, where evaluation scripts that are used for this task can also be found.

7 Evaluation of the Best in Context task

7.1 Assumptions

In this task, systems are required to return a ranked list of best entry points (one per article) to the user, representing the point in the article where they should start reading. The aim of the task is to first identify relevant articles (the fetching phase), and then to identify the element corresponding to the best entry points for the fetched articles (the browsing phase). In the fetching phase, articles should be ranked according to their relevance. In the browsing phase, we have a single element whose opening tag corresponds to the best entry point for starting to read the relevant information in the article.

7.2 Evaluation measures

Runs for the Best In Context (BEC) task were evaluated with two metrics:

1. A set based measure, BEPD (For BEP-Distance).
2. An extension of precision recall (EPRUM).

Both metrics use a base score for an element x, which is defined as 0 if x does not appear in a relevant document, i.e. a document containing a Best Entry Point (BEP). Otherwise, there exists a BEP b in the x's document and the measure, between 0 and 1, is defined as

$$s(x, b) = A \times \frac{L}{A \times L + d(x, b)}$$

where

- $d(x, b)$ is the distance (in number of characters) between the beginning of element x and the beginning of element b;
- L is the average document length (in characters)
- $A > 0$ is a parameter

Note that high values of A (e.g. 10) tend to give a score of 1 to any answer in a relevant document, hence the score does not discriminate whether x is near to or far from the BEP b. Whereas low values of A (e.g. 0.1) favour runs that return elements very close to a BEP.

BEPD The BEPD metric is the sum of all the single scores $s(x, b)$ over elements x of the run divided by the total number of best entry points. The measure is then averaged over runs (i.e. queries).

EPRUM-BEP-Exh-BEPDistance The EPRUM metric is an extension of precision recall suited for structured corpora and fine-grained user models. This metric is described in [8]. While standard precision-recall assumes a simple user model, where the user consults retrieved elements (elements returned by the retrieval system) independently, with EPRUM, we can capture the scenario where the user consults the context of retrieved elements. This is modelled with a parameter, which is the probability that a user goes from a returned element x to a BEP b.

EPRUM metric is defined by three parameters:

Targets (BEP) What are the targets, i.e. the (here the Best Entry Points).

Target relevance (Exh) What is the relevance of the target (here, fixed to the exhaustivity of the document which is always the maximum since we have only 1 exhaustivity level)

User behaviour (BEPDistance) How to compute the probability that the user goes from one element in the list to a BEP. In the context of the BEC task, this probability is simply defined as $s(x, b)$ for any BEP b. This behaviour is defined stochastically, that is we only know that the user has seen the BEP with probability $s(x, b)$.

Precision at recall r is defined as the ratio, for the user to achieve a recall r, of the minimum expected search length for the ideal run to the run's minimum expected search length.

Precision at rank k is defined as the expected search length (for the ideal run) for a user to achieve the same recall as the one achieved by the evaluated run divided by k.

In both cases the ideal run is the list of BEP. Both definitions reduce to the classical precision and recall when the standard user model is assumed, where the parameters (i.e. the probabilities) are either 0 or 1.

7.3 Results

Runs were evaluated with a parameter A equal to 0.01, 0.1, 1, 10, 100. Reported measures were:

- BEPD
- EPRUM precision recall graph
- EPRUM precision averaged over all recall values

8 Conclusions

9 Acknowledgments

References

1. C. Clarke, J. Kamps, and M. Lalmas. INEX 2006 retrieval task and result submission specification. In N. Fuhr, M. Lalmas, and A. Trotman, editors, *INEX 2006 Workshop Pre-Proceedings*, pages 381–388, 2006.

2. K. Järvelin and J. Kekäläinen. Cumulated Gain-based evaluation of IR techniques. *ACM Transactions on Information Systems (ACM TOIS)*, 20(4):422–446, 2002.
3. G. Kazai and M. Lalmas. eXtended Cumulated Gain Measures for the Evaluation of Content-oriented XML Retrieval. *ACM Transactions on Information Systems (ACM TOIS)*, 24(4):503 – 542, October 2006.
4. J. Kekäläinen and K. Järvelin. Using graded relevance assessments in IR evaluation. *Journal of the American Society for Information Science and Technology*, 53(13):1120–1129, 2002.
5. J. Kekäläinen and K. Järvelin. Using graded relevance assessments in IR evaluation. *Journal of the American Society for Information Science and Technology*, 53:1120–1129, 2002.
6. M. Lalmas. INEX 2005 retrieval task and result submission specification. In N. Fuhr, M. Lalmas, S. Malik, and G. Kazai, editors, *INEX 2005 Workshop Pre-Proceedings*, pages 385–390, 2005.
7. J. Pehcevski and J. A. Thom. HiXEval: Highlighting XML retrieval evaluation. In *Advances in XML Information Retrieval and Evaluation: Fourth Workshop of the INitiative for the Evaluation of XML Retrieval, INEX 2005, Dagstuhl Castle, Germany, November 28-30, 2005, Revised Selected Papers*, volume 3977 of *Lecture Notes in Computer Science*, pages 43–57, 2006.
8. B. Piwowarski and G. Dupret. Evaluation in (xml) information retrieval: Expected precision-recall with user modelling (eprum). In *SIGIR 2006: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.
9. B. Piwowarski, P. Gallinari, and G. Dupret. Precision Recall with User Modelling: Application to XML retrieval. *Submitted for publication*, 2005.
10. T. Tombros, B. Larsen, and S. Malik. The interactive track at INEX 2004. In N. Fuhr, M. Lalmas, S. Malik, and Z. Szlavik, editors, *Proceedings of the 3rd Workshop of the INitiative for the Evaluation of XML retrieval (INEX), Dagstuhl, Germany, December 2004*, 2005.

Appendix A: EvalJ

EvalJ is a java project[9], which implements the XCG metrics (in the XCGEval package), which were used to evaluate the Thorough and Focused tasks, and the PRUM metrics, which were used to evaluate the Best in Context task.

Running XCG

To run the XCG metrics (XCGEval package in EvalJ) from the command line use:

```
java -Dorg.xml.sax.driver=gnu.xml.aelfred2.XmlReader
     -jar jars/EvalJ.jar [-e] [-q] [-gnu/jfree]
     -config yourconfigfile.prop
```

In addition use -Xmx$SIZEm to increase memory allocation to $SIZE Mb.

The options are as follows:

[9] https://sourceforge.net/projects/evalj/

– Use -q to print evaluation results for each query. If not specified, only scores averaged over the query set are output.
– Use -e to score empty result topics as 0, i.e. topics that occur in the recall-base but for which no results were returned by a run. If not specified, only topics that occur in the run are scored.
– Use -gnu to produce gnuplot dat and gp files, or use -jfree to use the jfree graphics modules to output graphs as png files. If not specified, no graph data is output (this reduces run time and memory requirements).

To run XCG requires several parameters, which define how e.g. the measures to be used, the runs to evaluate, etc. These parameters are read at run time from the config file. The official parameter settings for the Thorough and Focused tasks at INEX 2006 are given below.

Thorough task. The official parameter settings in the config file for the evaluation of the Thorough task are:

```
TASK: Thorough
METRICS: ep/gr
OVERLAP: off
QUANT_FUNCTIONS: gen
ASSESSMENTS_DIR: $aPath/2006_assessmentsv4/
SUBMISSIONRUNS_DIR: $aPath/2006_runsv1/*
RESULTS_DIR: $aPath/2006_results
```

where $aPath is a placeholder for a folder name.

Focused task. The official parameter settings for the Focused task at INEX 2006 are as follows:

```
TASK: Focused
METRICS: nxCG
ALPHA: 1.0
DCV: 5, 10, 25, 50
OVERLAP: on, off
QUANT_FUNCTIONS: gen
ASSESSMENTS_DIR: $aPath/2006_assessmentsv4/
SUBMISSIONRUNS_DIR: $aPath/2006_runsv1/*
RESULTS_DIR: $aPath/2006_results
```

where $aPath is a placeholder for a folder name.

EPRUM and BEPD

In order to construct evaluate BEC runs, a database containing the assessments must be constructed. This is done following these three steps:

1. Create a database directory

2. Add the wikipedia collection
3. Add the assessments

The documentation (README file) contained in the EvalJ package contains further information on how to perform these steps.

The configuration file that should be used for Best In Context is the following.

```
<metrics>

 <EPRUM id="eprum-bep-100" only="BestInContext">
  <binary value="false"/>
  <generator value="BEPTargetGenerator"/>
  <quantisation value="Exh"/>
  <behaviour value="BEPDistance"><A value="100"/></behaviour>
</EPRUM>

 <EPRUM id="eprum-bep-10" only="BestInContext">
  <binary value="false"/>
  <generator value="BEPTargetGenerator"/>
  <quantisation value="Exh"/>
  <behaviour value="BEPDistance"><A value="10"/></behaviour>
</EPRUM>

<EPRUM id="eprum-bep-1" only="BestInContext">
  <binary value="false"/>
  <generator value="BEPTargetGenerator"/>
  <quantisation value="Exh"/>
  <behaviour value="BEPDistance"><A value="1"/></behaviour>
</EPRUM>

<EPRUM  id="eprum-bep-0.1" only="BestInContext">
  <binary value="false"/>
  <generator value="BEPTargetGenerator"/>
  <quantisation value="Exh"/>
  <behaviour value="BEPDistance"><A value="0.1"/></behaviour>
</EPRUM>

<EPRUM  id="eprum-bep-0.01" only="BestInContext">
  <binary value="false"/>
  <generator value="BEPTargetGenerator"/>
  <quantisation value="Exh"/>
  <behaviour value="BEPDistance"><A value="0.01"/></behaviour>
</EPRUM>

<BEPD only="BestInContext" id="bepd-100"><a value="100"/></BEPD>
<BEPD only="BestInContext" id="bepd-10"><a value="10"/></BEPD>
<BEPD only="BestInContext" id="bepd-1"><a value="1"/></BEPD>
```

```
 <BEPD only="BestInContext" id="bepd-0.1"><a value="0.1"/></BEPD>
 <BEPD only="BestInContext" id="bepd-0.01"><a value="0.01"/></BEPD>

</metrics>
```