Evaluation in (XML) Information Retrieval: Expected Precision-Recall with User Modelling (EPRUM)

Benjamin Piwowarski
Yahoo! Research Latin America
Santiago, Chile
bpiwowar@yahoo-inc.com

Georges Dupret
Yahoo! Research Latin America
Santiago, Chile
gdupret@yahoo-inc.com

ABSTRACT

Standard Information Retrieval (IR) metrics assume a simple model where documents are understood as independent units. Such an assumption is not adapted to new paradigms like XML or Web IR where retrievable informations are parts of documents or sets of related documents. Moreover, classical hypotheses assumes that the user ignores the structural or logical context of document elements and hence the possibility of navigation between units. EPRUM is a generalisation of Precision-Recall (PR) that aims at allowing the user to navigate or browse in the corpus structure. Like the Cumulated Gain metrics, it is able to handle continuous valued relevance. We apply and compare EPRUM in the context of XML Retrieval – a very active field for evaluation metrics. We also explain how EPRUM can be used in other IR paradigms.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation*

General Terms

Measurement, Theory

Keywords

Evaluation, Recall-precision, metric, XML Retrieval, Passage Retrieval, Web Retrieval

1. INTRODUCTION

Evaluation has always been a key problem in Information Retrieval (IR). The experimental practice started with the work on the Cranfield collection [4] and has been popularised since then by the TREC conferences and other IR challenges. State of the art evaluation metrics are used to compare different systems and to justify theoretical and/or pragmatical developments of IR systems. As a part of the evaluation process for emerging IR fields like XML or Web IR, developing adequate metrics is both an essential and an open question.

Among the different measures and criteria that have been proposed, standard metrics are most often combinations of recall and precision [1]. *Recall* is the proportion of retrieved elements among the relevant ones. *Precision* is the proportion of relevant elements among the retrieved ones. In standard or Web IR, the elements coincide with documents, while in the context of XML IR, the elements are the XML elements. As search engines results are usually ordered lists, precision is often computed *at a given cut-off value* or *at a given recall*. In this work, we use the term precision in this latter acception and refer to a basic retrieval unit as an "element".

All the metrics rely either explicitly or implicitly on assumptions concerning the nature of the collection, the notion of relevance and the user behaviour. Among others, precision-recall graphs make the following assumptions: (1) The user consults a ranked list ordered by decreasing relevance score; (2) Relevance of an element is binary and independent of the relevance of other elements; (3) The user has only access to one element at a time.

While assumption (1) is acceptable, the last two assumptions lead to contradictions. If a section has several paragraphs among which one is relevant, it is itself relevant to a certain extent, although less than its relevant paragraph. This contradicts assumption (2). In this paper, we formally introduce a concept that supersedes the classical notion of relevance, the so-called "idealism" [14, 11]. Assumption (3) is not adapted to structured corpora. Indeed, it seems natural to reward an element that gives access to relevant material through structural navigation (*e.g.* from a section to one of its paragraph) or through hyperlinks. We see that the *context* of an element, defined as the set of elements that can be reached through navigation from that element, plays a crucial role: The user is not restricted anymore to the retrieved element, *but to its context.*

We believe EPRUM is a substantial improvement over current metrics. First, EPRUM is formally derived from a explicitly defined user model: Computed values can be fully interpreted, user experiments can be straightforwardly used to set the metric parameters and EPRUM presents no analytical inconsistencies. Second, EPRUM user model is flexible. In the context of XML IR, navigation is not limited to browsing from an XML element to its descendants or ancestors. Third, the returned list may be composed of complex objects and is not restricted to a simple list of elements. An example of the usefulness of this property is the "Fetch and Browse" task of the Initiative for the Evaluation of XML Retrieval[1] (INEX), where each rank is composed of several pointers to elements within a given document. Eventually, with respect to standard PR, EPRUM formally includes graded relevance.

The plan of this paper is as follows. In Section 2, we show how we redefine the classical concepts of precision at a given recall. In Section 3, we describe how to extend the concept of relevance and

[1]http://inex.is.informatik.uni-duisburg.de/

detail the user model. An example of EPRUM computation is given in Section 4 while EPRUM formulae are derived in Section 5. We eventually compare EPRUM with other metrics in Section 6 and perform some experiments in Section 7.

2. EPRUM DEFINITION

The classical recall and precision metric [19] mentioned in the introduction does not accommodate well a complex user model and leads to several inconsistencies. Two main causes can be identified: Firstly, elements can have a relevant element in their context and the user can navigate to them. Secondly, navigation within the corpus structure can lead the user to see more than one relevant element for a single consulted list item. We propose here a generalisation of precision-recall, somewhat similar to [15], that can handle these problems.

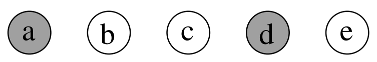
As in the classical definition, the recall value r is the number of relevant elements the user wants to see. The recall level ℓ ($0 < \ell \leq 1$) is defined as the ratio of a recall r to the total number t of relevant units. Our generalisation then relies on the definition of the minimum number of ranks m the user needs to consult in the list in order to reach a recall level ℓ, or said otherwise a recall value of ℓt.

The user starts considering the first rank of the list. If he finds more than ℓt relevant elements at this rank, then his information need is satisfied and he stops. In this case, the user effort has been restricted to the consultation of the first rank of the list (m is 1). If not, he proceeds to the second rank, etc. The definition of precision is based on the comparison of two minimum values: The minimum rank that achieves the specified recall over *all* the possible lists and over the *evaluated* list. For a given recall level ℓ, we define precision as:

$$\text{Precision@}\ell = \mathbb{E}\left[\begin{array}{c}\text{Achievement}\\ \text{indicator}\\ \text{for a recall } \ell\end{array} \times \frac{\begin{array}{c}\text{Minimum number of}\\ \text{consulted list items for}\\ \text{achieving a recall } \ell \text{ over all lists}\end{array}}{\begin{array}{c}\text{Minimum number of}\\ \text{consulted list items for achieving}\\ \text{a recall } \ell \text{ over the evaluated list}\end{array}}\right]$$

where the achievement indicator is used to set the precision to 0 if the recall level cannot be reached for the evaluated list. This is compatible with the classical definition of precision *at a given recall* where the precision is set to 0 if the list does not contain enough relevant elements: In this case, the user never reaches the specified recall level ℓ. This is equivalent to setting the search length (of the evaluated list) to infinite. For the minimum over all the lists, we suppose that there is at least a list for which the recall level can be achieved.

We illustrate this definition in the context of traditional IR where the user cannot navigate and where units are independent. Consider the next list where relevant units are represented in grey:



The standard definition of precision assigns a precision of respectively 1, 0.5 and 0 for recalls of 1, 2 and 3 (or more). Our new definition leads to the same values (this example being deterministic, the mathematical expectation symbol can be dropped):

At recall 1 The minimum number of elements the user has to consult, over all possible lists, is 1. The first element the user sees in the evaluated list is relevant and consequently the precision is 1.

At recall 2 The minimum number of elements the user has to consult, over all possible lists, is 2. For the evaluated system, the user needs to consult the list until –that is, he has to consult 4 elements. Precision is 0.5.

A recall of 3 cannot be attained by the user, the achievement indicator is 0 and hence precision is 0. As shown in this example, the new definition of precision-recall coincides with the standard definition. The interest of this formulation is that it accommodates more complex user and relevance models. It is possible to prove that, using the final formula of EPRUM and setting its parameters so as to mimic the standard user behaviour in traditional IR, we get the same result as TREC precision-recall at natural recall points. For an arbitrary level ℓ, results will differ since we use a different interpolation procedure. This procedure will be justified latter in the paper.

3. USER AND RELEVANCE MODEL

Current metrics suppose that a user sees the elements in their order of appearance in the result list. EPRUM on the other hand considers these elements as entry points to the collection from where the user can navigate to find relevant elements if he feels that this strategy is promising. EPRUM relates directly the user satisfaction to the number of distinct relevant elements he reaches. If an element of interest was discovered earlier by the user, the system is not rewarded. In this Section, we describe in detail the user model and illustrate it with some practical examples. We also present an important simplifying assumption that is needed to compute EPRUM and that has an impact on the user model we chose.

3.1 Idealism

In this section, we offer a distinction between the concepts of relevance and idealism. Although relevance is subject to many debates [17], we will suppose here that it is a well-defined property like is customarily done for other metrics as described in [14]. In order to handle multi-graded relevance, we also introduce a so-called multi-graded satisfaction level whose implications are described at the end of the section.

We first refine and extend the concept of relevance. In new IR paradigms, retrievable units cannot be considered independent like in traditional IR: For example, in XML IR, if a paragraph is relevant for a given information need, then its enclosing section also bears some relevance. In order to distinguish the intrinsic relevance of the paragraph from the "inherited" relevance of the section, we say that although the two elements are relevant, only the paragraph is *ideal*. By definition, an ideal element is always relevant but the reverse is true only in classical IR.

Ideal elements, unlike relevant elements, are supposed to be independent. Note that this hypothesis is similar to the independence of document relevance in classical IR: Ideal elements, as documents, can overlap *conceptually* (they can contain the same answer to an information need) as long as they do not overlap physically. In XML IR, this implies that ideal elements cannot be nested. This assumption is common among the new XML IR metrics [14, 11].

It is a complicated matter to identify a set of ideal elements as user agreement on which elements to include is usually low. In standard IR evaluation, using a graded relevance has already been proposed [12] in order to credit IR methods for their ability to retrieve highly relevant documents. In structured IR, it is important to distinguish the degree of idealism of different elements since they can be of significantly different value to the user: An element in XML can range from a mere paragraph to a whole section or document.

We define the probability $P(x \in \mathfrak{I})$, where $\mathfrak{I}$ is the set of ideal elements, as the percentage of users that regard element x as ideal. We introduce a threshold called the satisfaction level S which is a uniform random variable over [0,1] and we define the ideal set $\mathfrak{I}_s$ corresponding to a given satisfaction level s as the set of elements for which $P(x \in \mathfrak{I}) \geq s$. We can now define a sequence of decreasing thresholds s, to which corresponds a sequence of ideal sets $\mathfrak{I}_s$. By definition, if an element is ideal for a given satisfaction level, it remains ideal for all lower satisfaction levels: This means that $\mathfrak{I}_s \subseteq \mathfrak{I}_{s'}$ iff $s \geq s'$. It is a simple matter to integrate or sum the relevant formulae over decreasing satisfaction levels. As the number of ideal elements t can change for different satisfaction levels, the recall r can also change for the same ℓ. This explains why the TREC interpolation procedure cannot be applied directly.

Another implication is that a satisfaction level is associated with each user: users that are satisfied by elements that satisfy 40% of the users are also satisfied by elements that satisfy 50% of the users. This simplification is necessary to compute EPRUM, and could be changed by another way of clustering ideal elements together. In the remaining of the paper, we consider that the ideal set is deterministic and we drop the s subscript.

3.2 Navigating user model

In this Section, we define a generic user model that can be instantiated for different IR paradigms, though we still use XML IR as an example. We then define the notion of *consideration* (the user "clicks" on a link in the result list) and of *navigation* (the user navigates, within the corpus structure, from a given list item). We also introduce the concept of *discovering* an element, *i.e.* of seeing it for the first time. This is important since the user will only be satisfied when he *discovers* an ideal element. Seeing it a second time or more is not rewarded by EPRUM.

The ordered set of results presented to the user who issues a query is a list of pointers to hopefully ideal elements. It is called the list. We suppose that the list is *totally* ordered to simplify the formal development. Contrarily to standard IR, *an item in the list is not a document* but provides an access to some parts of the corpus. In Web IR, this would be one or more links to web documents, along with a surrogate. In XML IR, this can be a pointer to an XML element, or it can be a list of pointers to a set of elements from the same document, like for example the table of content for a chapter.

In standard IR, the user is assumed to *consider* every "document" of the list. In fact, the document is a surrogate that can be reduced to a simple link to the document: The user is then supposed to systematically follow that link. While this hypothesis holds in many cases, some new tasks make it unrealistic. In INEX for instance, the Fetch & Browse task consists in retrieving a list of sets of elements grouped by document in response to a given query. A list item in this task points to a set of elements belonging to the same article. It would be unreasonable to suppose that the user follows the link to *every* element of each article: It seems intuitive that the larger the set of elements, the less likely the user considers each of them – this is even more true if previously seen elements are not ideal. In Web IR, as pages are sometimes grouped by site, a probability of consideration conditional on former considered elements could be used to model a user that follows only some of the links. The *consideration* concept also suggests that, besides relevance, a valuable information would be whether an element makes the user "feel" that he could find some relevant information in the same document (XML IR) or in the same web site (Web IR).

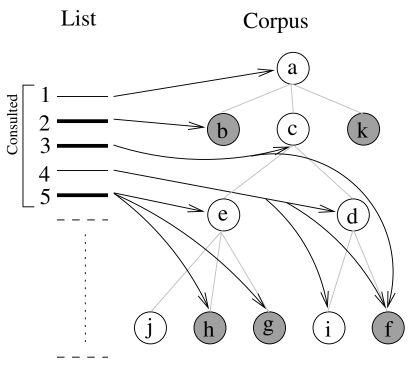


Figure 1: The ideal elements have a grey background. The user consults the first 5 ranks of the list, 3 of them (bolder lines in the list) leading the user to discover (or more) ideal elements. Elements are all within a same document whose structure is shown with light lines: a is the root node and has three children (b, c and k), etc. Arrows show the browsing behaviour of one particular user. An arrow from the list means that the user *browsed to* the element from a given rank.

From a considered element, the user can navigate using the corpus structure. Links or simple navigation in the document can lead the user to see elements which are not necessarily directly pointed by the list. The *context* of an element is defined as the set of elements that can be reached through navigation from it and includes the element itself. The exact coverage of the context depends on the collection and the existence of relations between elements. User navigation at a point in the list is restricted to the context of the corresponding entry element, which in turn is determined by the document structure. To model the user behaviour inside the context, EPRUM relies on a set of probabilities on simple events of the form "navigating from a list item to an element in the corpus". In the case of the XML paradigm for example, the probabilities of navigating from a rank to the parent, a descendant or a sibling of a pointed element can be set to values estimated by any adequate method. When the user is over with this exploration, he proceeds to the next entry of the list and repeats the process until his information need is satisfied.

We also introduce two closely related concepts: We say that a user *sees* an element when he navigates to it from another element or from the list, and that he *discovers* an element if he sees it for the first time. The distinction between "discovered" and "considered" is important for EPRUM because the system is rewarded only when elements are discovered.

We illustrate the model on the simple example of Fig. 1. In this figure, all the elements belong to a unique document and the context of each of these elements is the whole document. A possible user session is described next. We note by F_i the number of ideal elements the user has seen while examining the i first items of the list. (1) He browses to element **a**. Since element **a** is not ideal F_1 is naught and the user continues to consult the list; (2) He consults element **b** but again no element in its context. The element is ideal and $F_2 = 1$; (3) He consults **c** and this time explores the context, discovering element **f** which is ideal: $F_3 = 2$; (4) He consults **d**, decides to explore its context and discovers **f** and **i**. Element **i** is not ideal, while **f** is ideal *but* has already been seen by the user: F_4 still equals 2. (5) He consults **e** and discovers **g** and **h** in its context; both elements are ideal and have not been previously seen, hence $F_4 = 4$.

This example covers different aspects of our user model. Steps (1) and (2) are similar to the standard user model. Step (3) illustrates how the user can navigate to an ideal element. In step (4), a unique ideal element is retrieved for the second time. Because no new information is discovered, F remains equal to 2. Eventually, step (5) shows how an element can act as an entry point to more than one ideal unit.

We introduce now the probabilistic events used to model the user behaviour. We say that a rank is *consulted* only if the user reached this rank when searching for ideal elements. An element is *seen* if it was reached through navigation from a consulted rank. We denote $x \in S_k$ the fact that the element has been seen by a user that consulted the list up to rank k. We denote $\mathrm{P}(k \rightsquigarrow x)$ the probability that the user reaches an element x from rank k. The probability $\mathrm{P}(k \rightsquigarrow x)$ may depend on the query, the list returned by the evaluated system, the document collection structure (links, document structure, etc.) and the elements relevance. It is important to note that this probability takes into account all the physical steps taken by the user in order to go from a rank k to element x: For example, if from rank k the user can go to a web page z which in turn has a link to x, this is reflected in the probability $\mathrm{P}(k \rightsquigarrow x)$. The set of elements x for which $\mathrm{P}(k \rightsquigarrow x) > 0$ is the *context* of rank k.

We now illustrate how this probability can be estimated. Suppose we have recorded the navigation of several users from a rank k between three elements x, y and z as depicted in Fig. 2.

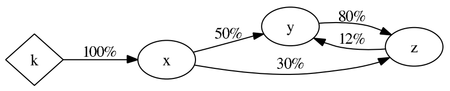


Figure 2: Example of user behaviour on a hypothetical document. The arrows depict the probability of navigation between elements: 50% of the users navigated from x to y and 80% of them continued to z. 20% of the users did not navigate to other elements from x.

Navigation from k would then be summarised by the following probabilities: $\mathrm{P}(k \rightsquigarrow x) = 1$, $\mathrm{P}(k \rightsquigarrow y) = 1 \times 0.5 + 1 \times 0.3 \times 0.12 = .536$, and $\mathrm{P}(k \rightsquigarrow z) = 1 \times .3 + 1 \times 0.5 \times 0.8 = 0.7$.

3.3 Main simplifying hypothesis

In this section, we present the main simplifying hypothesis necessary to derive a computable formula for EPRUM. We discuss and illustrate its implications.

HYPOTHESIS 3.1. *A user consults the context of a rank k independent of his previous navigation: The set of events $k \rightsquigarrow x$ are mutually independent. Formally, for any set of couples $\{(k_i, x_i)\}$, we have $P(\bigwedge_i k_i \rightsquigarrow x_i) = \prod_i P(k_i \rightsquigarrow x_i)$.*

This hypothesis means that even if the user navigates from *e.g.* a rank k to a paragraph, it will not give an indication on whether the user navigates from the rank k' to the second paragraph. To assess the impact of this hypothesis on the user model, we illustrate it on the example of Fig. 2. Without the simplifying hypothesis, to compute the probability $\mathrm{P}(k \rightsquigarrow y \wedge k \rightsquigarrow z)$ that the user navigates from rank k to both y and z, we would need to add the probabilities of the two possibilities: From k, the user navigates to x and then first to y and then to z ($0.5 \times 0.8 = 0.4$), or first to z and then to y ($0.3 \times 0.12 = 0.036$). The resulting probability is 0.436. If we accept the simplifying hypothesis, we can multiply the probability of navigating to y (navigating first by z or not) and to z (navigating first by y or not): $\mathrm{P}(k \rightsquigarrow y) \times \mathrm{P}(k \rightsquigarrow z) = 0.7 \times 0.536 = 0.3752$. The difference is significant, but less than if we considered only the direct navigation from x to y and z. In this case, the probabilities $\mathrm{P}(k \rightsquigarrow y)$ and $\mathrm{P}(k \rightsquigarrow z)$ would be respectively equal to 0.5 and 0.3, yielding a joint probability $\mathrm{P}(k \rightsquigarrow y \wedge k \rightsquigarrow z)$ of 0.15 only.

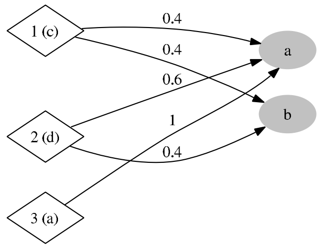


Figure 3: Rank 1 points to element c. Users can navigate to element a (or b) with a probability of 0.4. Element a is pointed to by rank 3.

We see that the simplifying hypothesis captures the fact that a user can first browse from rank k to x, and then to y. Moreover, it simplifies greatly the computation of the probability of the event $x \in S_k$ that a user sees an element x after he consulted the k first ranks of the returned list. We compute this probability here to illustrate the implication of the hypothesis. The result will be reused in the section about the EPRUM computation. The event $x \in S_k$ is true iff there is at least one rank from which the user browsed to x. Formally,

$$x \in S_k \equiv \bigvee_{j=1\ldots k} j \rightsquigarrow x$$

The hypothesis 3.1 implies that all causes (the user browsed to x from a given rank k) are independent of each other in terms of their abilities to influence the effect variable (the user has seen x). This is similar to the "noisy-or" hypothesis [8] which is often used in probabilistic models when an event is the consequence of many causes. Formally, we can show that under hypothesis 3.1 the probability of the event $x \in S_i$ becomes:

$$\mathrm{P}(x \in S_k) = 1 - \prod_{j=1}^{k} (1 - \mathrm{P}(j \rightsquigarrow x)) \tag{1}$$

This leads to a realistic model whose complexity is linear in terms of the number of elements leading to x. We illustrate the computation of Eq. (1): Suppose the first three ranks of the returned list lead respectively 40%, 60% and 100% of users to see the ideal element **a** as depicted in Fig. 3. After rank 1, the probability that a random user sees **a** is $1 - (1 - 0.4) = 0.4$. After the second rank, the probability is $1 - (1 - 0.4)(1 - 0.6) = 0.76$ and after the third, it becomes $1 - (1 - 0.4)(1 - 0.9)(1 - 1) = 1$. Consequently, 100% of users would see **a** after they consult the first three ranks while this would be the case of 76% of them after rank 2. Without the "noisy or" assumption, the above probability would require the knowledge of the interaction of the different navigations from the three first ranks to **a**. In practice, the number of such interactions grows exponentially with the number of elements in the context and lead to intractable complexity.

It is easy to show that such a user and relevance model encompass the classical IR ones: It is sufficient to set the idealism to be binary, and the navigational probabilities $\mathrm{P}(k \rightsquigarrow x)$ to 1 if rank k points to x and 0 otherwise. Acquiring the navigational probabilities needed by EPRUM is possible through the use of a parametric user model whose parameters would be tuned in order to reduce the discrepancy between the observations and the model predic-

Table 1: This table summaries the different situations that might happen with the settings described in the Section 4. The upper part of the table shows the rank where either a or b is first seen. The line labelled PS (Probability of the Scenario) gives the probability (expressed in % of users that would act the same) that such a situation is observed. The line labelled A/ML gives the ratio of the achievement indicator (0 if the recall cannot be reached) to the minimum search length (set to 3 if the recall cannot be achieved). The $\mathbb{E}$ column gives the expected values of A/ML over the different configurations.

	Table of scenarios									
rank	**Discovered ideal elements**									
1 (c)	**a,b**	**a**	**a**	**b**			**b**			
2 (d)		**b**		**a**	**a,b**	**a**		**b**		
3 (a)							**a**	**a**	**a**	**Sum**
PS (%)	16.0	9.6	14.4	14.4	8.64	12.96	9.6	5.76	8.64	*100%*
	Search for one ideal element									$\mathbb{E}$
A/ML	1	1	1	1	$\frac{1}{2}$	$\frac{1}{2}$	1	$\frac{1}{2}$	$\frac{1}{3}$	0.81
	Search for two ideal elements									$\mathbb{E}$
A/ML	1	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{3}$	$\frac{1}{3}$	0	0.37

tions. An example of a parametric model would be $P(k \rightsquigarrow x) = (1 + e^{\theta d(k,x)})^{-1}$ where $d(k,x)$ is the distance (in number of words) between the element pointed to at rank k and x, and θ is the user model parameter.

4. EPRUM KEY EXAMPLE

In this Section, we illustrate the computation of precision with the EPRUM metric on the small example illustrated in Fig. 3. The given information need has two ideal answers, the elements **a** and **b**. For simplification, we suppose that these elements are highly ideal and each rank of the returned list is a pointer to an element in the corpus. Probabilities are set as represented in the figure.

We want to evaluate the performance of an engine that returns a list that points to elements (**c**, **d**, **a**) in that order. The Table 1 gives a summary of the different probabilities of navigation. The first column of the upper table corresponds to the consulted list. The second column evaluates to 0.16 the probability that a user navigates to elements **a** and **b** from the first rank of the list returned by the engine, i.e. from a pointer to **c**. The third column represents the case where a user discovers element **a** navigating from the first rank and element **b** when navigating from the second rank. The probability of such a scenario is 0.096 because

$$\begin{aligned}&\text{P(navigates from 1 to } \mathbf{a}) \times \text{P(does } \textbf{not} \text{ navigate from 1 to } \mathbf{b})\\ &\quad \times \text{P(navigates from 2 to } \mathbf{b}) = 0.4 \times 0.6 \times 0.4 = 0.096\end{aligned}$$

The other configuration probabilities are computed likewise. We omitted from the table impossible scenarios like for instance a user that does not see **a** up to rank 3 (It contradicts the probability 1 of seeing **a** at rank 3). The probabilities over all scenarios naturally sum to one.

The two tables below the scenarios show the inverse search length multiplied by the achievement indicator (**A/ML**). The column $\mathbb{E}$ reports the inverse expected number of ranks the user has to consult to reach a given recall value (1 or 2 ideal elements). Its value is computed by summing the products of the terms of lines **A/ML** and **PS**. We can suppose that the minimum search length is respectively of 1 and 2 over all the possible lists for recall 1 and 2 respectively. Using the formula of Section 2, precision at recall 1 is $1 \times 0.81 = 0.81$ and at recall 2 it equals $2 \times 0.37 = 0.74$. These values are far superior to the standard precision-recall estimates (0.333 and 0). This is the expected EPRUM behaviour since the first two ranks can lead the user to see both ideal units.

5. COMPUTING EPRUM

In this section, we derive EPRUM, starting from the definition of Section 2 and using the user and relevance models described in the Section 3.

We first introduce some new notations. ML_ℓ is the minimum number of list items the user needs to consult if he requires ℓ% of the ideal units. N is the length of the list and A_ℓ is the "achievement" indicator that is equal to 0 if the user cannot reach the recall level ℓ. We use the superscript * to denote the variables related to the ideal list.

The rank of the list after which the user has just discovered ℓ% of the t ideal units is the minimal rank k for which F_k is greater or equal to the level ℓ multiplied by the total number of ideal units t:

$$\text{ML}_\ell = k \Leftrightarrow F_k \geq \ell t \wedge F_{k-1} < \ell t \tag{2}$$

Note that by definition before the first rank no ideal units have been found, *i.e.* $F_0 = 0$. We first have to suppose that the user behaviour is independent for the evaluated and the ideal lists:

$$\mathbb{E}\left[\text{A}_\ell \times \frac{\text{ML}^*_\ell}{\text{ML}_\ell}\right] = \mathbb{E}\left[\text{ML}^*_\ell\right]\mathbb{E}\left[\frac{\text{A}_\ell}{\text{ML}_\ell}\right]$$

Let us first consider the ideal case. Using the definition of a mathematical expectation and Eq. (2), we can write:

$$\mathbb{E}\left[\text{ML}^*_\ell\right] \quad = \quad \sum_{k>1}^{N^*} k\,\text{P}\left(F^*_k \geq \ell t \wedge F^*_{k-1} < \ell t\right) \tag{3}$$

where N^* is the length of the ideal list. The ideal list(s) depends on the specific user model instantiation and its computation can be more or less complex. Note that although there might exist more than one ideal list, it is possible to define one ideal list for any given recall r – and hence for any recall level ℓ.

Let's take an example to illustrate this point: Let **a**, **b**, and **c** be three elements; **b** and **c** are ideal. Like in Section 4, the list is composed of ranks reduced to one pointer to an element in the corpus. The probability of navigating from a pointer $\widehat{\mathbf{x}}$ to the pointed element **x** is 1. The probability of navigating from $\widehat{\mathbf{a}}$ to **b** (or **c**) is 0.9. For a recall 1, an ideal list would be a simple list restricted to one of the ideal elements, $\widehat{\mathbf{b}}$ or $\widehat{\mathbf{c}}$, with an expected length of 1. For a recall of 2, an ideal list would be $(\widehat{\mathbf{a}}, \widehat{\mathbf{b}}, \widehat{\mathbf{c}})$ because 81 % of the users would see two ideal elements after the first rank, 9 % after the second and 10 % after the third – thus implying an expected search length of 1.29.

When the number r of ideal elements the user requires is fixed and if we assume that the ideal list is known, it is possible to compute Eq. (3). As the number of discovered ideal elements never decreases ($F_k < \ell t$ implies $F_{k-1} < \ell t$), we have $\text{P}(F^*_k < \ell t \wedge F^*_{k-1} < \ell t) = \text{P}(F^*_k < \ell t)$. We also suppose that there always exists an ideal list that gives an access to r ideal elements, i.e. that there is a rank $k(r)$ for which $F^*_{k(r)} \geq r$. We can then rewrite the equation as:

$$\begin{aligned}\mathbb{E}\left[\text{ML}^*_\ell\right] &= \sum_{k\geq 1} k\left(\text{P}(F^*_{k-1} < \ell t) - \text{P}(F^*_k < \ell t)\right)\\ &= 1 + \sum_{k\geq 1} \text{P}(F^*_k < \ell t) \qquad (4)\end{aligned}$$

For the system list, we have to take into account the fact that the user might not be satisfied after having consulted all the list items, that is, he did not see ℓt ideal elements. This happens if $F_N < \ell t$: The ratio $\frac{A_\ell}{ML_\ell}$ is equal to 0 and thus this case can be ignored in the computation of the expectation. Otherwise, the ratio $\frac{A_\ell}{ML_\ell}$ can take the value $\frac{1}{k}$, for any $k = 1 \dots N$, with a probability $P(F_k \geq \ell t \wedge F_{k-1} < \ell t)$. Following a similar reasoning as for the ideal list, we can compute the expectation for the evaluated list:

$$\begin{aligned} \mathbb{E}\left[\frac{A_\ell}{ML_\ell}\right] &= \sum_{i=1}^{N} \frac{1}{k}\left(P(F_{k-1} < \ell t) - P(F_k < \ell t)\right) \\ &= 1 - \frac{1}{N}P(F_N < \ell t) - \sum_{k=1}^{N-1} \frac{1}{k(k+1)} P(F_k < \ell t) \quad (5) \end{aligned}$$

To estimate the two EPRUM main formulae, Eq. (4) and (5), we need to know how to compute the probability $P(F_k = f)$ that a user discovered f ideal elements. We first decompose it in simpler events related to the individual ideal elements. The user discovered f ideal elements with a probability:

$$P(F_k = f) = P\left(\sum_{x \in \mathfrak{I}} [\![x \in S_k]\!] = f\right) \quad (6)$$

where $[\![\text{event}]\!]$ is 1 (resp. 0) if the event is true (resp. false). It is straightforward to see that the sum $\sum_{x \in \mathfrak{I}} [\![x \in S_k]\!]$ equals f if there exists a subset $A \subseteq \mathfrak{I}$ of cardinality f for which $x \in S_k$ if *and only if* $x \in A$. We can then rewrite Eq. (6) into:

$$P(F_k = f) = \sum_{\substack{A \subseteq \mathfrak{I} \\ |A| = f}} P\left(\bigwedge_{x \in A} x \in S_k \wedge \bigwedge_{x \in \mathfrak{I} \setminus A} x \notin S_k\right)$$

Using hypothesis (3.1) that implies the independence of the $x \in S_k$ events, we can express the above expression as:

$$P(F_k = f) = \sum_{\substack{A \subseteq \mathfrak{I} \\ |A| = f}} \prod_{x \in A} P(x \in S_k) \prod_{x \in \mathfrak{I} \setminus A} P(x \notin S_k) \quad (7)$$

where each term is given by Eq. (1). This sum can be computed in a time quadratic in the number n of elements that were not seen with a probability of one (i.e. elements x such that $0 < P(x \in S_i) < 1$). When n is large enough (experiments have shown that $n = 10$ is enough) the sum can be approximated using a normal law. This can be justified by the Lindenberg-Feller central limit theorem [2]. We don't describe the method here due to space constraints.

EPRUM can now be computed using Eq. (4), (5), (7) and (1).

6. RELATED WORKS IN XML IR

We compare EPRUM with metrics proposed in XML IR, a very active community for evaluation metrics. INEX is the meeting point of this community where a corpus of XML documents, queries and their associated relevance assessments is constructed since 2002. In the INEX ad-hoc collection, documents have an explicit tree-like structure – typically they are composed of a header, followed by a body composed of several sections, etc. One of the major tasks is to retrieve, in answer to a given user's information need, a list of elements of the right granularity (a paragraph, section, etc). The inherent structure of the corpus has important consequences on the evaluation metrics suitability.

Important issues for (XML) IR metrics are near misses and overlap. It is generally admitted [11] that near misses, *i.e.* elements close to an ideal, should be rewarded, but less than exact answers, because they can act as entry points leading to one or more ideal elements. Early attempts [7, 9] to generalise the PR metric for XML IR tried to reward near misses by assigning "some" relevance to the elements nearby an ideal one. The limits of such a strategy for XML IR is now well known [14, 11]: Adding elements (the near misses) to the so-called "recall base" implies that systems that return only ideal elements don't achieve a 100 % recall. Both ideal elements and near misses have to be returned to achieve such a recall.

These generalisations of PR for XML IR are commonly considered to be "overlap positive" – using the terminology proposed by [21], which means that they reward system for retrieving twice the same ideal element, either directly or indirectly, and that the total reward for that ideal element increases with the number of times it is retrieved. Overlap neutral and negative is metrics are defined accordingly.

Another interesting metric is Tolerance To Irrelevance (T2I) [20] proposed for Video and XML IR. The core of this proposal is a user model considering overlap and near misses. The user consults the returned elements until he finds relevant information or until his tolerance (counted in number of words for example) to irrelevant material is reached. In both cases, he proceeds to the next element in the list. The model presents however some theoretical inconsistencies when the same ideal element can be reached from two elements in the list and precision is computed under the assumption that an element leads to *at most* one ideal element. Moreover, the relevance score of an element drops from one to zero depending on a preset number of words that precede it, which seems too strict a behaviour for our purpose. Note that the T2I user model can be viewed as a particular instantiation of the EPRUM user model.

The xCG (eXtended Cumulated Gain) metric family [10] was in 2005 the official INEX metric. xCG has been derived into two other metrics, namely nxCG (normalised xCG) and EP/GR (effort-precision/gain-recall). The latter metric has a definition which is close to EPRUM, as it is defined as the ratio of minimum lengths. We compare EPRUM with EP/GR in the next section. The xCG metrics partially manage to overcome the above mentioned precision-recall limitations while maintaining a simple definition. With xCG metrics, near misses can be taken into account without expanding the recall base: As for the EPRUM relevance model, xCG relies on the definition of a set of non overlapping ideal elements. The main difference between xCG metrics and EPRUM is that the latter is derived from a formal user model whereas this model is implicit in the former. This has some consequences on the stability of the metric and also causes some inconsistencies in the metric: For EPRUM the "gain" is only related to the discovery of an ideal element – an element can only be discovered once, which makes it naturally overlap neutral. For xCG metrics, the gain is not a well defined concept, especially for near-misses elements.

7. EXPERIMENTS

To assess a metric, several issues need to be addressed. First, one can show that the metric measures what it should measure. The best demonstration would rely on user experiments, but such data is not yet available. One can partly justify it through the theory underlying the metric: In EPRUM a well defined user model gives insight on what is measured. The faithfulness of the measure can also be proved experimentally. Using real system runs would be of no use since their behaviour is not sufficiently controlled. Another approach taken by [14] is to use synthetic runs constructed in a controlled way. Second, a metric should be stable and sensitive [3, 18]: What is the confidence on a difference in the evaluation of

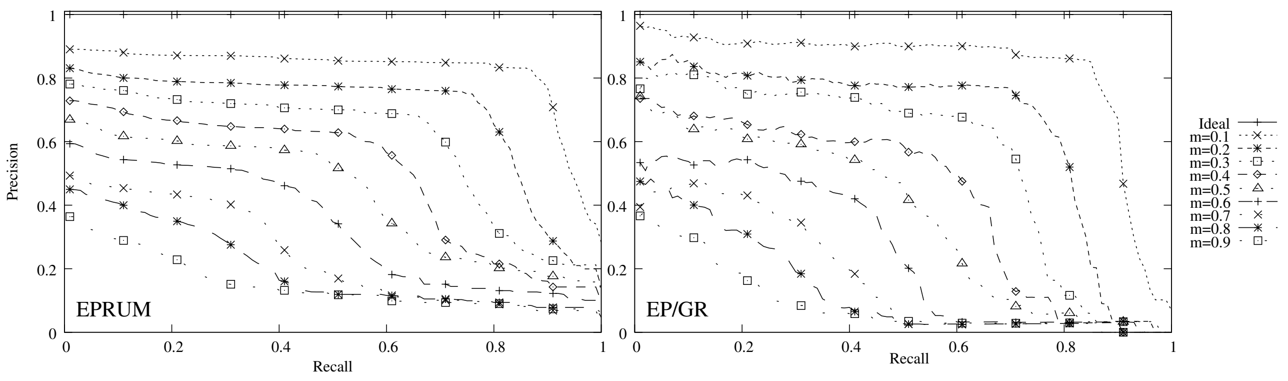


Figure 4: Comparison of EPRUM and xCG (EP/GR) with synthetic runs. The randomisation level of the ideal run varies from $m = 0.1$ to 0.9.

two or more systems and consequently on the superiority of one of them?

In this Section, we investigate only the first issue and underline the fact that xCG gives different results while its implicit user model is similar, and its ideal set of elements exactly identical.

In order to illustrate the metric behaviour on real world data, we performed some experiments using the INEX 2005 dataset. We used a simple model where each rank k is composed of a link to an element x_k in an XML corpus. Within that corpus, the only action that a user can make is going up or down in the XML tree. This is similar to most of the implicit user models proposed in XML IR evaluation, and in particular to the official INEX 2005 xCG metric we compare EPRUM with. We heuristically set the navigational probabilities as follows to the ratio $\text{length}(x_k)/\text{length}(y)$ if y contains x_k, $\text{length}(y)/\text{length}(x_k)$ if x_k contains y, and 0 otherwise– where the length is the number of characters the element contains. The intuition is that the more the elements share common content, the more likely the user is going to browse from one to another. (Other, more realistic models exists for XML as for example [20]). The other metric parameters were set so that the two metrics be as close as possible.

With respect to this user model, generating an ideal run from the ideal list of elements is quite simple: It is sufficient to use a list composed of the ideal elements ordered by decreasing probability of idealism. The collection we used is composed of the 29 ad-hoc topics of the INEX CO+S task along with their associated relevance assessments. From each of these assessments we generated an ideal run as described in [10]: Each of these runs is "ideal" for both xCG and EPRUM (with the user model described above) and does not contain any overlapping element. We then created nine new runs from it by applying some random transformation. For a given element of the list, we choose a direction (up or down) with an equal probability. Starting from this element in the XML structure, we repeat the following process with a probability $1-m$ of stopping at each loop (m is the probability to "move"): If going up (resp. down), we replace the current element by its parent (resp. one of its children). The list element is then replaced by the current element we have moved to. The whole process is done for every element of the list. We chose to move only along the ancestor-descendant axis because of the chosen user model.

In Fig. 4, we plotted EPRUM recall-precision graphs and xCG EP/GR graphs for m varying from 0.0 (ideal run) to 0.9 with 0.1 steps. We expected that the run performance would degrade smoothly from 0 to 0.9 (with a perfect performance at $m = 0$). This is the case for both EP/GR and EPRUM. The main difference between the two evaluation measures is the stability at different recall points of EPRUM over EP/GR. The curves with the higher randomisation (0.7 to 0.9) get a near null precision for a recall superior to 0.6 for EP/GR, whereas this is not the case for EPRUM. This could be explained by a difference in the user model, but it is also intrinsic to the xCG metrics where a given recall is achieved or not, whereas in EPRUM it can be achieved by a given percentage of the users. This also explains the smoother decrease of the different curves.

We performed some further test comparing xCG and EPRUM. We used 45 runs from the INEX 2005 participants and the 29 topics of the "CO.Focused task" which aims at retrieving the non overlapping ideal answers to an information need. We computed the mean average precision (MAP) for every topic of every run, leading to a total of 645 values. We first compared MAP of EP/GR and EPRUM. The Pearson's correlation coefficient was of 0.75. A more detailed analysis showed that 48 top ranked values are highly correlated (around 0.9) while it is not the case for others (0.65). This implies that even though they share a common user model and the same ideal base, EPRUM and EP/GR behave differently.

Other user behaviour are realistic but not represented, even implicitly, in the xCG metrics. If we compare a model where an ideal element is replaced by one of its siblings, for example, EPRUM user model could be easily extended so as to observe the same smooth degradation of the PR curves, while for xCG the extension is not straightforward. This is an important difference because we believe that systems that return a nearby paragraph instead of the ideal paragraph should be rewarded, especially in a context where users disagree on which elements are ideal and where ideal elements can be small in size.

8. DISCUSSION AND CONCLUSION

We presented a new metric based on a generic user model. EPRUM reduces to standard precision recall (PR) when its parameters are set to mimic the classical IR model. We distinguished the notion of idealism from the notion of relevance. We showed how to compute the precision at a given recall level. Note that even if we don't present the formulae in this paper, the standard precision at a given document (or rather: rank) cut-off value can be estimated likewise.

EPRUM can be easily used in new IR paradigms, like XML IR, Web IR, Video IR, and passage retrieval by setting the user model parameters accordingly. For Video IR and passage retrieval, EPRUM can be set to a T2I-like user model which is quite adapted and corresponds to some common assumptions about the user behaviour [20]. In Web IR, we could use the Page-Rank hypothesis [13]: A user will follow an hyperlink with a probability inversely proportional to the number of links in the consulted Web Page.

In the context of XML IR, EPRUM is a natural extension of xCG and T2I. Being well-defined, it resolves some of the inconsistencies present in these metrics. Moreover, EPRUM can be easily extended to new user models. Result lists are not restricted to a mere list of document or element surrogates but can include list of elements grouped by articles like the Fetch and Browse task of INEX.

For all metrics save T2I, which shares the same definition as EPRUM, the interpretation of the evaluation results is not straightforward. EPRUM reports the discovery of ideal elements, which has a clear semantic. At a given recall level, or at a given rank, EPRUM is the average amount of average extra effort, measured in number of list items, that the user has to consent in order to discover as many ideal elements as an ideal system. An interesting extension would be to consider time instead of list items as a controlling factor.

Another important contribution of EPRUM is that items in the returned list can be complex objects. For instance, each item could be a list of elements belonging to the same document. For the other metrics the evaluation of such a task is not as straightforward because the list can only be composed of element while the EPRUM definition only relies on the navigation from list items and allow a natural evaluation of such a task.

With respect to standard IR, EPRUM introduces naturally the concept of graded relevance which was one of the focus of Cumulated Gains (CG) metrics [12]: For EPRUM, a judgement between 0 and 1 corresponds to the percentage of users that, given the same information need, would judge relevant the document. EPRUM could thus be used as a natural extension of precision-recall in IR when graded judgements are used. More investigation is needed to compare the behaviour of EPRUM with CG metrics.

For classical IR, ranking documents by decreasing probability of relevance is optimal [16]. With respect to the new IR paradigms, this raises the question of whether we could separate the search task in two: First identify the set of ideal elements and then compute how to present the results to the user. Search engines would be in charge of the first task, while the second task would be the realm of metrics.

Future work includes user experiments to determine the parameters for some XML IR tasks and experiments to investigate thoroughly the EPRUM stability and its differences with other metrics. We also plan to work on specifying more in detail what are the conditions on the user model that should be enforced to generate a single ideal list, or on how to compute (or approximate) the minimum search length over all possible lists without computing the ideal list.

9. REFERENCES

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Addison Wesley, New York, USA, 1999.

[2] P. Billingsley. *Probability and Measure*. Wiley, New York, 1979.

[3] C. Buckley and E. M. Voorhees. Evaluating evaluation measure stability. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 33–40, New York, NY, USA, 2000. ACM Press.

[4] C. Cleverdon. The cranfield tests on index language devices. In *Aslib proceedings*, volume 19, pages 173–192, 1967.

[5] N. Fuhr, M. Lalmas, and S. Malik, editors. *INEX 2003 Proceedings*, 2003.

[6] N. Fuhr, M. Lalmas, S. Malik, and G. Kazai, editors. *INEX 2005 Proceedings*, 2005.

[7] N. Gövert, G. Kazai, N. Fuhr, and M. Lalmas. Evaluating the effectiveness of content-oriented XML retrieval. Technical report, University of Dortmund, Computer Science 6, 2003.

[8] D. Heckerman and J. S. Breese. A new look at causal independence. In *Proceedings of the Tenth Annual Conference on Uncertainty in Artificial Intelligence (UAI–94)*, pages 286–292, San Francisco, CA, 1994. Morgan Kaufmann Publishers.

[9] G. Kazai. Report on the INEX 2003 metrics group. In Fuhr et al. [5], pages 184–190.

[10] G. Kazai and M. Lalmas. Inex 2005 evaluation metrics. In Fuhr et al. [6].

[11] G. Kazai, M. Lalmas, and A. P. Vries. The overlap problem in content-oriented XML retrieval evaluation. In *Proceedings of the 27th annual international conference on Research and development in information retrieval*, pages 72–79, Sheffield, UK, July 2004. ACM Press.

[12] J. Kekäläinen and K. Järvelin. Using graded relevance assessments in IR evaluation. *JASIS*, 53(13):1120–1129, 2002.

[13] P. Lawrence, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[14] B. Piwowarski and P. Gallinari. Expected ratio of relevant units: A measure for structured information retrieval. In Fuhr et al. [5].

[15] V. V. Raghavan, G. S. Jung, and P. Bollmann. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions on Information Systems*, 7(3):205–229, 1989.

[16] S. E. Robertson. The probability ranking principle in IR. *Journal of Documentation*, 33:294–304, 1977.

[17] T. Saracevic. Relevance reconsidered. In *Proceedings of the Second International Conference on Conceptions of Library and Information Science*, volume 39, pages 201–218, Copenhagen, Danemark, 1996.

[18] E. M. Voorhees. Variations in relevance judgments and the measurement of retrieval effectiveness. In *SIGIR '98: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia*, pages 315–323. ACM, 1998.

[19] E. M. Voorhees. Common evaluation measures. In *The Twelfth Text Retrieval Conference (TREC 2003)*, number SP 500-255, pages 1–13. NIST, 2003.

[20] A. Vries, G. Kazai, and M. Lalmas. Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *Proceedings of RIAO (Recherche d'Information Assistée par Ordinateur (Computer Assisted Information Retrieval)), Avignon, France*, Apr. 2004.

[21] A. Woodley and S. Geva. Xcg overlap at inex 2004. In Fuhr et al. [6].