Inf Retrieval (2012) 15:558–590
DOI 10.1007/s10791-012-9192-1

Extended structural relevance framework: a framework for evaluating structured document retrieval

M. Sadek Ali · Mariano Consens · Mounia Lalmas

Received: 26 May 2011 / Accepted: 7 February 2012 / Published online: 6 March 2012

Abstract A structured document retrieval (SDR) system aims to minimize the effort users spend to locate relevant information by retrieving parts of documents. To evaluate the range of SDR tasks, from element to passage to tree retrieval, numerous task-specific measures have been proposed. This has resulted in SDR evaluation measures that cannot easily be compared with respect to each other and across tasks. In previous work, we defined the SDR task of tree retrieval where passage and element are special cases. In this paper, we look in greater detail into tree retrieval to identify the main components of SDR evaluation: relevance, navigation, and redundancy. Our goal is to evaluate SDR within a single probabilistic framework based on these components. This framework, called Extended Structural Relevance (ESR), calculates user expected gain in relevant information depending on whether it is seen via hits (relevant results retrieved), unseen via misses (relevant results not retrieved), or possibly seen via near-misses (relevant results accessed via navigation). We use these expectations as parameters to formulate evaluation measures for tree retrieval. We then demonstrate how existing task-specific measures, if viewed as tree retrieval, can be formulated, computed and compared using our framework. Finally, we experimentally validate ESR across a range of SDR tasks.

Keywords Tree retrieval · XML retrieval · Evaluation · Relevance · Redundancy · User navigation · Effectiveness measures

M. S. Ali (✉) · M. Consens
Bahen Center, University of Toronto, 5 King's College Circle, Toronto, ON, Canada
e-mail: sali@cs.toronto.edu

M. Consens
e-mail: consens@cs.toronto.edu

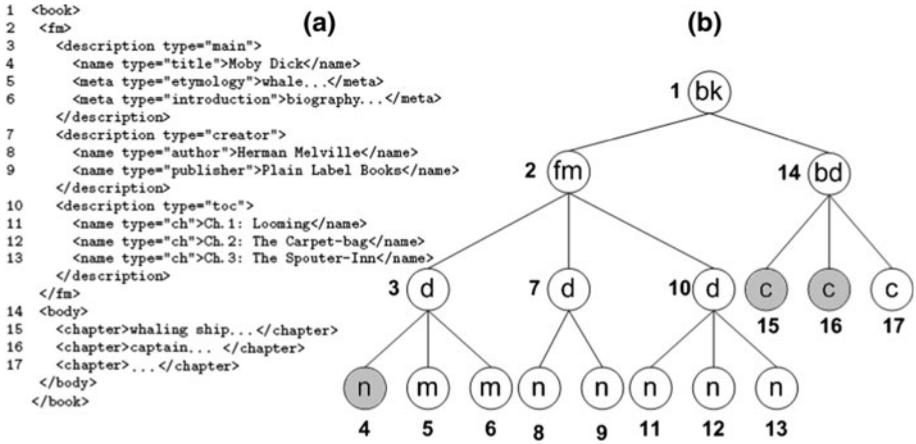
M. Lalmas
Yahoo! Research, Av. Diagonal 177, 8th floor, 08018 Barcelona, Catalonia, Spain
e-mail: mounia@acm.org

1 Introduction

Much of the work in document retrieval has focused on the goal of developing systems that retrieve relevant documents. In contrast, the goal of a structured document retrieval (SDR) system is to retrieve relevant parts of documents. We refer to document parts as *sub-document* results. SDR is particularly advantageous when dealing with long documents and those covering a wide variety of topics.

SDR systems exploit the structure of a document in two ways. First, referred to as a structural hint [40], sub-documents are ranked based on whether their encoding help users in locating relevant information. Second, referred to as a structural constraint [42], a user may direct the system to search for sub-documents with a desired encoding, using a query language such as NEXI [43] or XQueryFT [8].

Several types of sub-document results exist in SDR, each of them "modelling" how users locate relevant information. We illustrate these through examples taken from our earlier work [5]. Let a collection contain the extract of the book (formatted in XML) shown in Fig. 1a. The document structure of the book is, in this case, a tree, which is shown in Fig. 1b; the tags have been abbreviated as follows: `book` (`bk`), `front matter` (`fm`), `body` (`bd`), `description` (`d`), `name` (`n`), `meta` (`m`), and `chapter` (`c`). The line numbers of elements shown in Fig. 1a correspond to the node ID of each corresponding node in Fig. 1b. Consider the query "ship captain in Moby Dick". The query matches terms in different structural parts of the book extract; specifically, node 4 (match on "Moby Dick"), node 15 (on "ship") and node 16 (on "captain") in Fig. 1b. For a document retrieval task, a system returns root nodes to model the user accessing the whole book. For a focused retrieval task [25], as illustrated in Fig. 2a, an SDR system may return nodes (encoded as elements or text passages) at separate ranks, which provides the user with focused information but at the cost of having to examine results from the same book at multiple rank positions. For a tree retrieval task [5], as illustrated in Fig. 2b, an SDR system returns subtrees at separate ranks (the first rank corresponds to a subtree taken from the book extract), which provides the user with single results that can direct the user to one or more relevant parts of a book.



```
1  <book>
2   <fm>
3     <description type="main">
4       <name type="title">Moby Dick</name>
5       <meta type="etymology">whale...</meta>
6       <meta type="introduction">biography...</meta>
      </description>
7     <description type="creator">
8       <name type="author">Herman Melville</name>
9       <name type="publisher">Plain Label Books</name>
      </description>
10    <description type="toc">
11      <name type="ch">Ch. 1: Looming</name>
12      <name type="ch">Ch. 2: The Carpet-bag</name>
13      <name type="ch">Ch. 3: The Spouter-Inn</name>
      </description>
    </fm>
14  <body>
15    <chapter>whaling ship...</chapter>
16    <chapter>captain... </chapter>
17    <chapter>...</chapter>
    </body>
  </book>
```

Fig. 1 **a** Extract from a book in XML markup, **b** Tree structure of book with relevant nodes highlighted

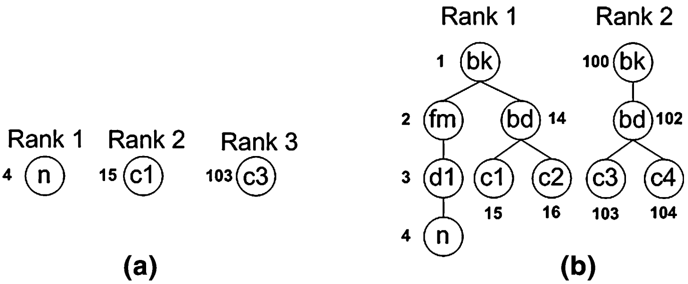


Fig. 2 Different SDR approaches; **a** element/passage, **b** tree

The evaluation of the effectiveness of a classical information retrieval (IR) system (such as a document retrieval system) is derived from the number of hits (relevant documents retrieved) and misses (relevant documents not retrieved) in the system output. In contrast, the output of an SDR system consists of *hits* (relevant sub-documents retrieved), *misses* (relevant sub-documents not retrieved), **and near-misses**. *Near-misses* are retrieved sub-documents that *may* not contain relevant information, but from which relevant information can be accessed via *navigation* e.g. a user browsing, scrolling down in the user interface, or following links. Therefore, SDR evaluation must take into account, not only the relevance of sub-documents, but the fact that users may navigate within documents to locate relevant information. The latter is usually not considered in classical (document) IR evaluation.

We refer to *user navigation* as the effort a user spends to locate relevant information from search results. We illustrate how user navigation can cause *redundancy*. Consider the ranked list in Fig. 2a from the book extract in Fig. 1a (nodes 4 and 15 at ranks 1 and 2, respectively). The system first returns node 4. Upon seeing node 4, the user might navigate to other nodes in the document. If the user saw node 15 by navigating to it from node 4 then he or she would experience what we refer to as *redundancy* when accessing node 15 directly at rank 2. SDR evaluation must account for how navigation can cause users to see relevant information more than once (redundantly).

Much of the existing work in SDR evaluation has been done in the context of the Initiative for the Evaluation of XML retrieval (INEX)[1], a collaborative and international effort dedicated to the development of effective XML or focused retrieval systems. Since 2002, INEX has investigated a wide range of SDR search tasks. This has resulted in task-specific evaluation approaches for element retrieval [24, 28, 34, 35], passage retrieval [31] and tree retrieval [5].

It is widely known that the evaluation of the range of SDR tasks has challenged INEX since its beginnings [41]. For instance, in our earlier work [5], we showed how most of the current approaches cannot evaluate tree retrieval because they are not able to represent how users satisfy their information need with tree-structured results. Analogous limitations have also been observed when customizing measures to evaluate specific search tasks [32]. This situation has resulted in SDR evaluation measures (and performance results) that cannot be compared with respect to each other and across search tasks. There are three main reasons for this: (1) current SDR measures have different ways to consider and calculate relevance, user navigation and redundancy, (2) they rely on task-specific assumptions of how the user information need is satisfied, and (3) they depend on the relevance assessment methodology.

[1] http://www.inex.otago.ac.nz/.

The main contribution of this work is to address the above limitations by proposing a single framework, called the Extended Structural Relevance (ESR) framework, that allows evaluation across SDR search tasks. ESR is related to our earlier work [5], where we show that tree retrieval is sufficient to capture all existing SDR approaches based on hits in the output. ESR extends our earlier work by considering not only hits, but also, near-misses and misses. More significantly, ESR revisits the relationship between relevance, user navigation and redundancy posited in our earlier work [5] to allow the development of measures that share the same set of parameters when evaluating SDR system performance. A *substantial* benefit is that it then becomes possible to compare the performance differences between SDR systems, where various models of user navigation and relevance are involved. The flexibility to support a wide variety of measures in a single framework is an important advancement in SDR for investigating future search tasks, where navigation has to be accounted for.

The outline of this paper is as follows. Section 2 reviews current approaches to SDR evaluation. Section 3 reviews tree retrieval and provides the notation for this work. Section 4 presents our ESR parameters for relevance, user navigation and redundancy. Section 5 presents the main contribution of this work, the Extended Structural Relevance (ESR) framework. Section 6 presents how to represent existing SDR evaluation measures in ESR. Section 7 presents experimental results comparing our ESR proposals to existing SDR measures. Finally, Sect. 8 concludes with remarks and future work.

2 Related work

The first SDR systems investigated in INEX were element retrieval systems. Their aim was to return relevant XML elements from a collection of XML documents as answers to a given query. The first measures used at INEX to evaluate element retrieval effectiveness consisted of adaptations of classical IR measures, where the notion of a document was replaced by the notion of an element. These early SDR measures considered the relevance of elements (as simple hits and misses), but ignored user navigation and redundancy. Later approaches to SDR evaluation proposed measures that capture user navigation and redundancy, as well as applying to other SDR tasks. We introduce some of the approaches next, while the actual measures are presented in Sect. 6.

Extended cumulated gain (XCG) [24] is a family of cumulated gain (CG) [20] measures for evaluating element retrieval. XCG is motivated on the observation that the effect of redundancy on the relevance of results is akin to wasted user effort because the same information, seen more than once, is not relevant to the user [27]. Effectiveness in XCG is defined by comparing the user gain in relevant information from a system to the gain obtained by spending the same effort in an ideal system. Ideal elements provide the best results for the user to see relevant information with the least effort. An ideal system ranks elements such that a user maximizes their information gain within a minimum number of ranks and experiences a minimum amount of redundancy. Kazai [23] noted two significant problems with respect to ideality. First, assessing ideal elements and an ideal ranking is a two-fold optimization, which is costly. Second, ideality introduces instability [11], stemming from the chosen assessment methodology determining what constitutes an ideal element.

Precision-Recall with User Modelling (PRUM) [35] is an extension of PRecall [37] where navigation to ideal elements is stochastic. PRUM measures precision based on the number of ranks in the output where the user obtains relevant information from ideal elements. Like XCG, PRUM requires knowing the ideal elements but, unlike XCG, it does not require an explicit ranking of ideal elements. The main contribution of PRUM is that it proposes a probabilistic model for user navigation that can be validated through studies of user behaviours. Its main drawback is that, like XCG, it is prone to instability, as it too relies on the adopted methodology for choosing the ideal elements.

A related measure, which solves some of the problems of PRUM by substantially reducing its complexity, and which also allows for graded relevance, is the measure of Expected Precision-Recall with User Modelling (EPRUM) [33]. For a given recall, it defines precision as a comparison between the minimum rank that achieves the given recall in an ideal system versus the minimum possible rank that achieves the given recall in the actual system. This approach, although simpler to calculate than PRUM, does not address instability because of its reliance, like PRUM, on ideality.

Highlighting XML evaluation (HiXEval) proposed in Pehcevski and Thom [31], and further finalized in Kamps et al. [22], was developed to evaluate the performance of systems that retrieve (or can be modelled as retrieving) passages, where a passage is a block of text, delineated or not with XML tags (when delineated, the passage is an XML element). We refer to this search task as passage retrieval. HiXEval measures are adaptations of classical IR precision and recall. Unlike XCG and PRUM, HiXEval does not rely on ideality. The main limitation of HiXEval is that it assumes that user navigation does not extend beyond the boundaries of retrieved passages and considers redundancy as only occurring between adjacent retrieved text passages that overlap each other. Overlap of text is a special-case of redundancy, and thus limits HiXEval in the investigation of the overall effect of redundancy when measuring system performance.

Structural Relevance (SR), proposed in our earlier work [5], evaluates tree retrieval. SR is a measure of the user expected gain in relevant information given that users may experience redundancy. SR does not rely on ideality. The main contribution of SR is that it proposes an integrated probabilistic model for expressing relevance, user navigation and redundancy. The key drawback of SR is that it is limited to the measurement of precision based on hits in the output.

User effort and redundancy have been investigated outside INEX. Salton et al. [39] investigated effort in passage retrieval in full-text search. Studies in web retrieval demonstrate how performance is improved by ranking results based on predictions of user navigation within web pages (either modelled from user clicks [17] or based on tracking navigation e.g. [1]). Other work includes Keskustalo et al. [29] who propose a relevance feedback mechanism based on simulating how users prefer to spend effort reading documents and providing feedback to the system to refine search results. In SDR, users see relevant information redundantly because of *information fragmentation*, i.e. documents are fragmented into sub-documents [30]. Redundancy has also been considered in search result diversification e.g. [14], which stems from the problem of *information duplication*, i.e. the same information appears in more than one document [9]. The aim is to rank documents to minimize the amount of redundant information contained in them [2]. Whereas our work focuses on the issue of redundancy in IR evaluation, research on diversification is concerned with the ranking of documents.

In the next section, we recall how tree retrieval can be used to model a range of search tasks (including element, passage, and document retrieval), and we introduce some notation. In Sect. 4, we describe how our proposed ESR framework extends SR to account for near-misses and misses.

3 Tree retrieval task

In our earlier work [5], tree retrieval is defined as the task of *"returning trees that provide the user with access to the document nodes in the collection that are relevant to the user's information need"*. For a given query, the system outputs a ranked list of trees. The user seeks relevant information in a retrieved tree by looking at the content contained in its nodes. While doing so, the user may navigate from the nodes in the tree to other parts of the document. At any point, the user may choose to return to the system output to access the next lower-ranked tree. This process continues until the user either satisfies his or her information need or exhausts the set of trees retrieved by the system.

Tree retrieval is a general task that can model many SDR search tasks. Figure 2a (shown in Sect. 1) illustrates an example of how trees can model document retrieval (by retrieving the root node of documents), element retrieval (by retrieving a node from the document), and passage retrieval (by retrieving either single nodes or trees of sibling nodes connected by their lowest-common ancestor node).[2]

The evaluation of tree retrieval tasks rests on the following three requirements originally posited in our earlier work [5]:

(i) the relevance of *retrieved* trees in the output are not independent of each other and depend on whether users tolerate *redundancy*,
(ii) the purpose of the system is to retrieve trees that afford a user access to relevant information by directly visiting a node in the tree or through *navigating* from a visited node into the rest of the document, and
(iii) the same relevant information may be expressed in trees of varying structure.

To illustrate how these requirements affect the evaluation of SDR systems, consider the trees in the ranked list shown in Fig. 3 as the output from an SDR system for the query "ship captain in Moby Dick" submitted by a user seeking literary references. First, the nodes in the trees in ranks 1 and 2 appear in the tree in rank 3. The user who sees all three trees will see each retrieved node at least twice (i.e., redundantly). This illustrates Requirement (i), in that, the relevance of the tree in rank 3 must account for all of its nodes having been *retrieved* earlier in the trees at ranks 1 and 2 and thus seen by the user. Second, from the tree in rank 1, the user may *navigate* to the nodes that appear in the later trees in both ranks 2 and 3. This illustrates Requirement (ii), in that, the relevance of these later trees will be affected by the user navigating from the nodes in the earlier tree. Third, the trees in ranks 2 and 3 would be relevant as literary references because they contain the same relevant chapters. This illustrates Requirement (iii), in that, the evaluation must account for relevant information being retrieved in trees of varying structure.

The requirements above present significant challenges for using current approaches to evaluate tree retrieval systems (discussed at length in our earlier work [5]). Classical approaches to evaluation assume that results are relevant independently of each other, which invalidates Requirement (i). In the context of SDR evaluation, HiXEval does not consider user navigation beyond retrieved passages so it invalidates Requirement (ii). Measures based on ideality (as proposed for PRUM and XCG) do not meet Requirement (iii), of encoding the same information in different trees. This is because it is not practical to determine all possible and equivalent ideal trees. In contrast, SR meets Requirements (i),

[2] In this paper, trees correspond to passages which match exactly with the boundaries of content in document nodes, which is sufficient to evaluate our proposed ESR framework for a number of INEX tasks (Sect. 7).

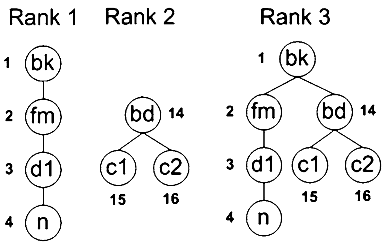


Fig. 3 Trees retrieved from the same document

(ii) and (iii) by using node-level assessments of relevance and user navigation to infer relevance, and by capturing user navigation and redundancy in tree-structured outputs without relying on ideality. But, SR is limited to measuring precision because it does not consider near-misses.

We now define the notation used in this paper. We denote the output of the tree retrieval task as a ranked list $R = t_1, t_2, \ldots, t_k$ of k distinct subtrees t_i from a collection C of trees. We denote the sublist of R up to rank i as R_i. The collection C is a forest of trees where each tree represents a document. A *tree* $T = \{T_V, T_E\}$ is a connected, directed, acyclic graph where T_V is a set of nodes, T_E is a set of edges between pairs of nodes from T_V.

Two subtrees from a collection are distinct if one contains a node not found in the other. Subtrees represent the sub-documents retrieved from the collection. A subtree $t = (t_v, t_e)$ of tree T in collection C satisfies $t_v \subset T_V$ and $(e_1, e_2) \in t_e$ if there is a path from nodes e_1 to e_2 in T. Moreover, when we refer to the tree t as a set, it refers to its set of nodes t_v. A subtree is a tree, and we use the terms interchangeably, unless stated otherwise.

The simplest tree is a single node called a *singleton*. We model element retrieval as systems that retrieve singletons. A singleton is a subtree t with a single node $t_v = \{e\}$ and no edges $t_e = \emptyset$. We refer interchangeably to subtrees with a single node as either singletons or nodes. A ranked list of nodes $R = e_1, e_2, \ldots, e_k$ is considered to be the same as a ranked list of singletons $R = t_1, t_2, \ldots, t_k$ where $t_i = \{e_i\}$. We differentiate between nodes (singletons) and trees using e and t, respectively. Specific to XML, we refer to nodes as *elements*. XML elements are nodes in the document tree of an XML document.

4 Relevance, user navigation and redundancy

Extended Structural Relevance (ESR) is a framework to calculate the user expected gain by conditioning the relevance of seen information with the probability of whether the information is both seen and not redundant to the user. Our framework encapsulates, as parameters, relevance, user navigation and redundancy, which we formalise in Sects. 4.1, 4.2 and 4.3, respectively.

4.1 Relevance

In IR evaluation, the *relevance* of information is a judgment made by a human assessor on whether the subject matter of the information is meaningful to a given information need. In classical retrieval, the relevance of information objects (e.g. documents) is assumed independent from each other. This is often not the case in SDR because users may navigate between sub-documents, and some information may be seen redundantly [44].

Table 1 Gain in SDR

Case	Gain	Description
Hits	Gain without effort	Retrieved tree seen once
Misses	No gain	Not retrieved tree not seen
Near-misses	Gain with effort	Not retrieved tree seen once

As posited in Piwowarski et al. [35], a user *gains* relevant information in SDR when it is *seen* by either retrieval, navigation, or a combination of both. However, a user may consider the information contained in the sub-document, albeit relevant, not useful, i.e. sub-optimal gain, because either it is redundant or its encoding format does not provide an ideal context [24]. In this paper, we refer to the gain from seeing a sub-document as a *relevance value*. In classical IR, because documents are assumed independent, relevance and relevance value coincide.

How to assess the relevance of sub-documents is an active area of SDR research [36]. Kazai [23] showed that the assessment methodology, based on ideality, introduces instability into the measures (discussed in Sect. 2). The author suggested that instability can be avoided by: (a) assessing the relevance of information independently of redundancy in the output, (b) assessing relevance without considering how a user may navigate to information, and (c) evaluating system effectiveness based on the effect of user navigation and redundancy on the user gain in relevant information. Suggestions (a) and (b) remove the need to assess ideality. Suggestion (c) implies that good SDR measures evaluate how users spend effort to achieve gain. In this work, we address suggestions (a) and (b) by assuming independence between relevance and user navigation (Assumptions 1 below). We address suggestion (c) by using expected gains and losses.

We recall from Sect. 1 that users gain relevant information from hits and near-misses. Near misses are defined in Kazai and Lalmas [24] as retrieved sub-documents that, may or may not be relevant, but which can be navigated from by the user to see unretrieved, relevant information. In this work, we reverse this definition. We consider a *near-miss* as a relevant sub-document that has not been retrieved and that can be accessed by the user via navigation from retrieved sub-documents. A *hit* is a relevant tree in the output. Finally, a *miss* is a relevant tree that is not seen by the user. These three cases define the basis of gain in ESR and are summarized in Table 1.

In ESR, we consider user navigation as a stochastic process. This is based on the observations in Hammer-Aebi et al. [19] where users see nodes by navigating via a graphical user interface from given nodes. Therefore, systems are evaluated in ESR based on *expected relevance values*, which are calculated by conditioning relevance value by the different cases (hits, misses, and near-misses) where relevant information is possibly both seen by the user and redundant to the user.

The *expected relevance value gain* is $E[rel(a)|a \textit{ is seen } \wedge \textit{ not redundant}]$ for both hits and near-misses. Relevant sub-documents that are not seen by the user are called *misses*, and we refer to the expected relevance value of a miss as a *loss*. For a miss, the expected relevance value loss is $E[rel(a)|a \textit{ is not seen}]$.[3] To calculate relevance value in ESR, we make the following assumptions:

[3] For completeness, we note that users experience relevance value loss when relevant information is seen redundantly, i.e., $E[rel(a)|a \textit{ is seen} \wedge \textit{redundant}]$. We refer to this as the *relevance value shrinkage*, in that, this value represents the diminution in the expected relevance value that the user can gain from hits and near-misses. For any tree, the total probability for expected relevance value is $P(a$ is seen, a is not

Assumption 1 (*Structural Relevance Assumptions*) Relevance is independent of how a user navigates to locate relevant information. Relevance is dependent on whether the user sees relevant information redundantly, i.e., more than once.

These assumptions are required to justify conditioning relevance value on the probability of whether information is seen and redundant. In essence, we are assuming the dependence of relevance on the outcome of the user spending effort (i.e. redundancy) and not dependent on the amount of effort spent (i.e. ranks consulted and user navigation). Using the above assumptions, we obtain $E[rel(a) | a\ is\ \ seen \wedge\ not\ redundant] = rel(a) \times p(a\ is\ \ seen \wedge\ not\ redundant)$. We next use this to develop expected relevance values across hits, misses, and near-misses to show how gain is calculated in ESR.

Consider a hit a_{hit} at a given rank m in a ranked list R. At rank m, the user sees the tree in the output. The tree will not be redundant to the user if it has not been navigated to from higher-ranked trees. Let $1 - p(a_{\text{hit}};R_{m\text{-}1})$ denote the probability that the user does not navigate to it from $R_{m\text{-}1}$ the higher-ranked trees. The expected relevance value gain from a hit is thus $E(a_{\text{hit}}) = rel(a_{\text{hit}}) \times (1 - p(a_{\text{hit}};R_{m\text{-}1}))$.

Next, consider a miss a_{miss} in a ranked list R. For it to be a miss, the user would not see it. So, the user would not navigate to it from the trees in the output R. Let $1 - p(a_{\text{miss}};R)$ denote the probability that the user does not navigate to see tree a_{miss}. Thus, the expected relevance value loss from a miss is $E(a_{\text{miss}}) = rel(a_{\text{miss}}) \times (1 - p(a_{\text{miss}};R))$.

Finally, consider the near-miss a_{nm} in a ranked list R. $p(a_{\text{nm}};R)$ denotes the probability that the user navigates from the trees in the output R to see tree a_{nm}. The expected relevance value gain from a near-miss is thus $E(a_{\text{nm}}) = rel(a_{\text{nm}}) \times p(a_{\text{nm}};R)$.

This completes our calculation of expected relevance values, namely:

1. $E(a_{\text{hit}}) = rel(a_{\text{hit}}) \times (1 - p(a_{\text{hit}};R_{m\text{-}1}))$, the expected relevance gain from a hit at rank m seen in the output,
2. $E(a_{\text{miss}}) = rel(a_{\text{miss}}) \times (1 - p(a_{\text{miss}};R))$, the expected relevance loss from a miss not seen in the collection,
3. $E(a_{\text{nm}}) = rel(a_{\text{nm}}) \times p(a_{\text{nm}};R)$, the expected relevance gain from a near-miss seen in the collection.

These expectations are crucial to our ESR framework. We will revisit these in Sect. 4.3 where we describe how to calculate *redundancy*, i.e. $p(a;R)$. In Sect. 5, these expectations form the basis of ESR. In Sect. 6, we then propose several SDR measures formulated using ESR. Finally, in Sect. 7, we test our proposed measures by evaluating a range of SDR tasks.

4.2 User navigation

A user may navigate from the retrieved results to seek (further) relevant information. Our interpretation of navigation is largely based on the work of Ali et al. [7], who define navigation as the effort spent by a user in seeking relevant information. More precisely, given a retrieved tree, a user may choose to seek from any node in that tree, via navigation, relevant information contained in nodes outside of that retrieved tree.

Footnote 3 continued
redundant) + $P(a$ is seen, a is redundant) + $P(a$ is not seen, a is not redundant) + $P(a$ is not seen, a is redundant) = 1. A tree cannot be both unseen and redundant, therefore $P(a$ is not seen, a is redundant) = 0. We do not further consider shrinkage in this paper.

To formally capture this, we introduce the *user navigation graph* which is a graph of a partition of the nodes in the collection where the edges of the graph are weighted. The user navigation graph allows modelling different navigational strategies in tree retrieval (e.g. navigation via document structure, contextual markup, semantic linking) at different granularities, as well as leading to faster computation [6, 7]. The weight between two nodes reflects the effort associated with the user navigating between them. Weights can be derived through the analysis of clicks, time spent, common routes, or retinal focus, and form the basis to calculate probabilities of user navigation in ESR.

Our methodology for measuring effort is inspired by the study in Hammer-Aebi et al. [19], where for a given information need, the user is tasked with finding, judging and marking the relevant parts of a retrieved document. The study begins by presenting the user with a document where retrieved information has been highlighted. The user's attention is directed to an initial highlight, referred to as the *entry point*. The user then navigates within the document using whatever means provided by the graphical user interface (such as scrollbars or hyperlinks in a table of contents). The user navigation is recorded as steps between nodes along a route starting from the entry point. The effort spent to make each step is measured. Examples of measured effort include cumulated gain [20], tolerance to irrelevance [44], expected search length [13], or time taken to read documents [15].

Let us now demonstrate user navigation along routes. Consider an XML document encoding an article (a) with sections (sec and ss1, respectively) and paragraphs (p). Given a user information need, the tree shown in Fig. 4a shows an article where nodes e_3 and e_4 are relevant. For our example, let us assume that the user navigates solely by clicking on hyperlinks such that a node is visited if and only if the user clicks on a link to the node. The node identifiers are shown beside each respective node, and their character lengths are shown in parentheses. So, for instance, node e_3 is 30 characters long. Figure 5 shows three examples of routes. Route 1 describes a user who entered the document via node e_3, then stepped to node e_1 then e_2 then e_4. Route 1 is composed of three steps; $e_3 \rightarrow e_1$, $e_1 \rightarrow e_2$, and $e_2 \rightarrow e_4$. As a possible measure of effort, let the number of times a step is observed indicate the ease with which users navigate it. Based on the routes shown in Fig. 5, step $e_3 \rightarrow e_1$ requires less effort than step $e_3 \rightarrow e_2$ because $e_3 \rightarrow e_1$ occurs twice whereas step $e_3 \rightarrow e_2$ occurs only once.

Next, to determine the probabilities needed to calculate ESR, we partition the nodes in the collection into a user navigation graph where directed edges are defined based on the routes users navigate. The weights on these directed edges are inversely proportional to the effort that users spend. To calculate ESR, we calculate probabilities for navigating steps based on the user navigation graph weighted by some function of effort. A higher probability for a step between two nodes corresponds to a lower effort for the user to take the

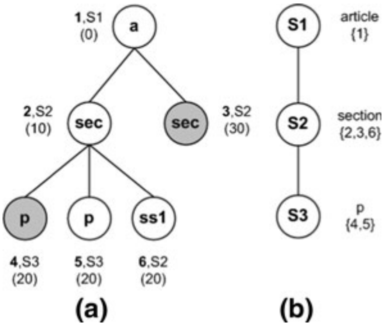


Fig. 4 **a** Tree structure of an article, **b** User navigation graph based on partition

Fig. 5 Examples of routes navigated

Route 1	$e_3 \rightarrow e_1 \rightarrow e_2 \rightarrow e_4$
Route 2	$e_3 \rightarrow e_2 \rightarrow e_4 \rightarrow e_5$
Route 3	$e_3 \rightarrow e_1 \rightarrow e_6$

Table 2 Elementary user navigation weights and navigation probabilities (in parentheses)

	e_1	e_2	e_3	e_4	e_5	e_6
e_1	0	1 (0.5)	0	0	0	1(0.5)
e_2	0	0	0	2 (1.0)	0	0
e_3	2 (0.66)	1 (0.33)	0	0	0	0
e_4	0	0	0	0	1 (1.0)	0
e_5	0	0	0	0	0	0
e_6	0	0	0	0	0	0

step. Let $\tilde{p}(e_i; e_j)$ denote the probability of navigating to node e_i from node e_j, i.e., $e_j \rightarrow e_i$. Thus, if $\tilde{p}(e_i; e_j) > \tilde{p}(e_a; e_b)$ then we can conclude that the user spends less effort to take step $e_j \rightarrow e_i$ than for step $e_b \rightarrow e_a$.

The most obvious partition to choose is the collection itself where the nodes in the user navigation graph correspond, one-to-one, with the nodes in the collection. We refer to this case as the *elementary user navigation graph*. Let our user navigation graph be the elementary case, which, in this case, is the tree shown in Fig. 4a with bi-directional edges. Let us consider the steps in the observed routes (such as the examples shown in Fig. 5) as edges in the user navigation graph.

Let $w(e_i;e_j)$ be the weight on a directed edge between node e_j to node e_i in the user navigation graph. For this example, let the weight $w(e_i;e_j)$ be the number of occurrences of step $e_j \rightarrow e_i$ in Fig. 5 and let us assume that the effort spent is independent across routes and steps. Thus, for instance, the weight $w(e_1;e_3)$ is 2 and $w(e_2;e_3)$ is 1 given the routes shown in Fig. 5[4] Table 2 summarizes our example weighting matrix for the elementary user navigation graph in Fig. 4a given the observed routes in Fig. 5.

We now calculate the *ESR navigation probabilities*. We calculate the probability of the user navigating the step $e_j \rightarrow e_i$ using $\tilde{p}(e_i; e_j) = w(e_i; e_j)/W(e_i)$ where $W(e_i) = \sum_{j=1}^{N} w(e_i;e_j)$ denotes the total weight of the directed edges in the user navigation graph leading to e_i. The total probability of navigating to node e_i from all other nodes in the collection is $\sum_{i \neq j} \tilde{p}(e_i; e_j) = 1$, if and only if it is possible for the user to navigate to e_i. Otherwise, $\tilde{p}(e_i; e_j)$ is 0. For instance, the probability $\tilde{p}(e_3; e_1)$ is $2/3 = 0.66$ because the total weight on the edges leading to e_3 is $w(e_3;e_1) + w(e_3;e_2) = 2 + 1 = 3$ and $w(e_3;e_1) = 2$. The values in parentheses in Table 2 summarize the ESR navigation probabilities modelled as elementary user navigation using the routes shown in Fig. 5.

In practice, the weights in elementary user navigation graphs cannot be determined (e.g. via human studies) because the graphs can be very large. Indeed, if N is the number of nodes in the collection, for each node e_i, there will be $N - 1$ probabilities $\tilde{p}(e_i; e_j)$ needed

[4] A special case of navigation is *abandonment* which occurs when a user opts to *not* navigate. We denote the probability that a user abandons navigation from node e with $\tilde{p}(e; e)$. To account for abandonment, we could include the terminal nodes of routes in our weights $w(e;e)$. For instance, consider the routes shown in Fig. 5, our weights for abandonment would be $w(e_4;e_4) = 1$, $w(e_5;e_5) = 1$, $w(e_6;e_6) = 1$ and $w(e;e) = 0$ for all other nodes. Other approaches would be to include in our weighting scheme the amount of time users spend in given nodes. For simplicity, we ignore abandonment in our examples so $w(e;e) = 0$ and $\tilde{p}(e; e) = 0$ for all nodes.

Table 3 Summary model weights and navigation probabilities (in parentheses) where $S1$, $S2$, and $S3$ are the summary nodes shown in Fig. 4b

	$S1$	$S2$	$S3$
$S1$	0	2 (1.0)	0
$S2$	2 (0.4)	1 (0.2)	2 (0.4)
$S3$	0	0	1 (1.0)

to define navigation. Therefore, $N \times (N - 1)$ weights are needed to calculate $\tilde{p}(e_i; e_j)$ for all nodes in the collection, which is impractical to assess in user studies for large N. We therefore consider a simplified model for navigation where the user navigates from one node subset to another, where the set of subsets form a partition of the nodes in the collection. For instance, the nodes in the tree shown in Fig. 4a can be grouped by their XML tags: `article` node e_1; `section` nodes e_2, e_3, and e_6; and `paragraph` nodes e_4 and e_5. Figure 4b shows a graph based on this partitioning scheme where $S1$ contains `article` nodes; $S2$ contains `section` nodes; and, $S3$ contains `paragraph` nodes. We proposed this approach originally in Ali et al. [6] using XML tags to partition nodes, and later validated our approach in Ali et al. [7].

Using this partition, we weight the directed edges between partitions. For instance, in Table 2, the outgoing steps from the nodes in $S2$ are: $e_3 \rightarrow e_1$ twice, $e_3 \rightarrow e_2$, $e_2 \rightarrow e_4$ twice. This corresponds to the following weights on the edges from $S2$: $w(S1;S2) = 2$, $w(S2;S2) = 1$, and $w(S3;S2) = 2$. Table 3 shows the resulting weighting matrix. We approximate the probability of navigation as $\tilde{p}(e_i; e_j) \approx \tilde{p}(S_i; S_j) = w(S_i; S_j)/W(S_i)$ where S_i denotes the partition of node e_i and S_j denotes the partition of node e_j, respectively. For instance, $\tilde{p}(e_3; e_1) \approx \tilde{p}(S2; S1) = w(e_3; e_1)/(w(e_2; e_4) + w(e_3; e_1) + w(e_3; e_2)) = 2/(2+2+1) = 0.4$. Table 3 summarizes (in parentheses) the ESR navigation probabilities for this model.

Partitioning schemes, such as the one described above, can be used to model different navigational strategies and lead to less costly computation. A further computational reduction is presented in our earlier work [5] where user navigation can be calculated over an infinite number of steps, i.e., using steady-state probabilities.

For instance, in Ali et al. [7], we use the partitioning schema developed in Consens et al. [12] and analysis from the eye-tracking study in Hammer-Aebi et al. [19] to model navigation in the INEX Wikipedia collection as a graph consisting of four partitions (namely, `article`, `section`, `ss1`, `other`) where each partition is weighted inversely to the tag depth of the nodes included in the partition. We refer to this as a *depth-weighted summary model of navigation* and use this model in our experiments (Sect. 7).

4.3 Redundancy

Redundancy occurs when a user sees the same relevant information more times than they tolerate [44]. Previously, in Sect. 4.1, we defined the user expected gain (and loss) of relevance value without clarifying how to calculate redundancy. In ESR, redundant information is considered not relevant to the user, and we consider information redundant if it is seen more than once.

Below, we restate the expected gains in Sect. 4.1 from hits (1) and near-misses (3), and the expected loss from misses (2).

$$E(a_{\text{hit}}) = (1 - p(a_{\text{hit}}; R_{m-1})) \cdot rel(a_{\text{hit}}) \tag{1}$$

$$E(a_{\text{miss}}) = (1 - p(a_{\text{miss}}; R)) \cdot rel(a_{\text{miss}}) \tag{2}$$

$$E(a_{\text{nm}}) = p(a_{\text{nm}}; R) \cdot rel(a_{\text{nm}}) \tag{3}$$

where R is a ranked list output of k trees, R_i is a sublist of R up to rank i such that if $i > k$ then $R_i = R$ and if $i \leq 0$ then $R_0 = \emptyset$, a_{hit} is a hit at rank m, a_{nm} is a near-miss, a_{miss} is a miss, $rel(a)$ is the relevance value of tree a, and $p(a;R)$ is the probability that the user will see tree a once by navigating from the trees in output R.

Now, we explain the calculation of $p(a;R)$ using the probability $\tilde{p}(t_i; t_j)$ that a user navigates from the nodes of tree t_j to the nodes in tree t_i. The user sees a tree by navigating to all of its nodes. The probability of seeing a tree by navigating to its nodes from a given tree is presented in our earlier work [5]. We state the probability here, as follows,

$$\tilde{p}(t_i; t_j) = \frac{\sum_{e_j \in t_j/t_i} \sum_{e_i \in t_i/t_j} \tilde{p}(e_i; e_j)}{|t_i| \cdot |t_j|} \tag{4}$$

where t_i is a tree from the collection, t_j is a different tree from the collection, x/y denotes the set of nodes in tree x not in tree y, e_i and e_j are nodes, $|t|$ is the number of nodes in tree t, and the probability $\tilde{p}(e_i; e_j)$ is the probability that a user will navigate to node e_i given that he or she navigates from node e_j (as shown in Sect. 4.2).

We next explain navigation between trees shown in (4). Consider a user navigating from the nodes in tree t_j to the nodes in tree t_i. Assume that each visit to a node by the user is independent. From a visit to node f in subtree t_j, the expected number of distinct nodes from subtree t_i that the user would see is $E[t_i;f] = \sum_{e \in t_i} \tilde{p}(e;f)$. For each node in t_j, the previous expected number of distinct nodes has a maximum value of $|t_i|$. We refer to t_j as the previous subtree, and t_i as the current subtree. The number of nodes seen in the current subtree from the previous subtree is $\sum_{f \in t_j} E[t_i;f]$. The maximum number of nodes seen is $|t_i| \cdot |t_j|$. The proportion of the nodes in the current subtree that were seen from the previous subtree is $\tilde{p}(t_i; t_j) = \sum_{f \in t_j} E[t_i;f]/(|t_j| \cdot |t_i|)$. This is the probability that the nodes in the current subtree have been seen from the previous subtree. Substituting the expected number of distinct nodes for $E[t_i;f]$, the probability becomes $p(t_i; t_j) = (\sum_{f \in t_j} \sum_{e \in t_i} p(e;f))/(|t_i| \cdot |t_j|)$ and, thus we obtain (4).

Finally, in Sect. 4.1 above, we defined redundancy in ESR as the conditioning probabilities for expected gains (losses) from hits and near-misses (misses). We recall that $p(a;R)$ denotes the probability $P(a \textit{ is seen once};R)$ that tree a is seen once by navigating from the trees in the output R. Assume that the navigation from each tree in the output is independent. We can calculate redundancy $p(a;R)$ using (4), as follows

$$p(a; R) = 1 - \prod_{j=1}^{k} (1 - \tilde{p}(a; t_j)) \tag{5}$$

where a is a tree in the collection, $R = t_1, t_2, \ldots, t_k$ is an output of k trees, and $\tilde{p}(a; t_j)$ is shown in (4).

4.4 Example toy system and models

In this section, we present a toy collection, a navigation model, a set of relevance judgments, and three example system outputs that we will use in later Sects. 5 and 6.5 to demonstrate ESR.

Table 4 Relevance value of assessments *rel*(e), for tree e in A

	e_3	e_4
Binary Relevance	1	1
Relevance by Length	30 characters	20 characters

Table 5 User navigation $\tilde{p}(e_i; e_j)$

	e_1	e_2	e_3	e_4	e_5	e_6
e_1	0	0.53	0.16	0.11	0.11	0.11
e_2	0.63	0	0	0.133	0.133	0.133
e_3	1	0	0	0	0	0
e_4	0.5	0.5	0	0	0	0
e_5	0.5	0.5	0	0	0	0
e_6	0.5	0.5	0	0	0	0

Table 6 Three example system outputs

	Ranks		
	1	2	3
System 1 ($R1$)	e_1	e_3	e_4
System 2 ($R2$)	e_1	e_2	e_6
System 3 ($R3$)	e_3	e_1	e_4

For a query, consider the retrieval of elements from the article shown in Fig. 4a. Let the assessed elements be e_3 and e_4, i.e. $A = e_3, e_4$. When relevance is binary, let $rel(e_3) = rel(e_4) = 1$. When we use the number of highlighted characters to measure relevance (relevance by length), let $rel(e_3) = 30$ and $rel(e_4) = 20$. The relevance values for both *binary relevance* and *relevance by length* of e_3 and e_4 are shown in Table 4. User navigation, the probability of the user navigating to a node from a given node[5], is shown in Table 5.

Table 6 shows the outputs for three systems for the query. System 1 ($R1$) retrieves a near-miss in rank 1 and hits in ranks 2 and 3. System 2 ($R2$) retrieves three near-misses. System 3 ($R3$) retrieves a near-miss in rank 2 and hits in ranks 1 and 3. We expect System 2 to have the worst performance because it does not retrieve any hits, and System 3 to have the best performance because it retrieves a hit in rank 1, whereas System 1 does not retrieve a hit until rank 2. We expect System 1 to be the second best performing system.

This completes the presentation of relevance, user navigation and redundancy in ESR. In the next section, we present the overall ESR framework where these expected gains and losses are calculated over the relevant trees in the collection in order to compute the user total relevance value gain (loss) for a given output composed of hits, misses, and near-misses, respectively.

[5] Our example ESR navigation probabilities are based on the PRUM user navigation model, used in Sect. 6.4, for hierarchical navigation which can be found in Equation 9 (p. 22) in Piwowarski et al. [35]. For the current example, we use the PRUM probabilities as weights on the edges of the article graph shown in Fig. 4a and obtain ESR user navigation as follows: $\tilde{p}(e_i; e_j) = p(e_j \rightsquigarrow e_i) / \sum_e p(e \rightsquigarrow e_i)$. There are, of course, other means to estimate $\tilde{p}(e_i; e_j)$. Indeed, our calculation of navigation is purely illustrative. Our intention is not to suggest that ESR and PRUM share equivalent navigation models (because they do not—see Footnote 10).

5 Extended structural relevance framework

Our proposed *Extended Structural Relevance* framework (ESR) provides the means to formulate measures based on the user expected gain (or loss) in relevance value given redundancy. ESR is motivated by the collection partitioning scheme presented in Bollman [10] (which, in turn, is largely motivated by the much earlier work in Robertson [38]). Bollman [10] shows that a family of document retrieval evaluation measures (such as precision, recall, and fallout) can be derived from the number of hits and misses in the output and the collection. The ESR framework represents a similar family of parameters for SDR measures based on partitioning sub-documents in the output and collection into hits, misses and near-misses.

We begin by partitioning the relevant trees in the collection into hits and misses in the output. For a given information need, consider the relevant trees in the assessments A that are hits in the output R. The hits are obtained from the intersection $R \cap A$ (6). The misses are obtained from the set difference A/R (7). The hits $A \cap R$ and misses A/R define a partition because $A/R \bigcap A \cap R = \emptyset$ and $A/R \bigcup A \cap R = A$[6].

$$\text{Hits} = R \cap A \tag{6}$$

$$\text{Misses} = A/R \tag{7}$$

If we assume that relevance value judgments are independent as stated in Assumptions 1, then the total expected relevance value for hits, misses, and near-misses can be obtained by summing the expectations across the trees in the appropriate set of judgments [(6) and (7), respectively]. The total expected relevance value gain from hits E[Hits, R, A] is obtained by summing the expected relevance value of the trees in the set *Hits* (6) using the expected relevance value gain for a hit in (1) (in Sect. 4.3). Similarly, the total expected relevance value loss from misses E[Misses, R, A] is obtained by summing the expected relevance value of the trees in the set *Misses* (7) using the expected relevance value loss for a miss in (2) (in Sect. 4.3). Finally, the total expected relevance value gain from near-misses E[Near-misses, R, A] is obtained by summing the expected relevance value of the trees in the set *Misses* (7) using the expected relevance value gain for a near-miss in (3) (in Sect. 4.3). Thus, the total expected relevance values can be stated, as follows,

$$E[\text{Hits}, R, A] = \sum_{a \in R \cap A} rel(a) \cdot (1 - p(a; R_{m-1})) \tag{8}$$

$$E[\text{Misses}, R, A] = \sum_{a \in A/R} rel(a) \cdot (1 - p(a; R)) \tag{9}$$

$$E[\text{Near-misses}, R, A] = \sum_{a \in A/R} rel(a) \cdot p(a; R) \tag{10}$$

where $R = t_1, t_2, \ldots, t_k$ is a ranked list of k trees, $A = a_1, a_2, \ldots, a_n$ is a set of n assessments, m is the rank of a in R, $rel(a)$ is the relevance value of tree a, and $p(a;R)$ is the probability that the user will see a once when consulting R, i.e., redundancy (5).

[6] Equations 6 and 7 are defined using perfect matches between the trees in the assessments (A) and the output (R). A more general, albeit more complex, approach is to define hits as the trees seen with certainty $p(a;R) = 1$, and misses as the trees not seen with certainty $p(a;R) < 1$. This is a practical issue and does not constitute any loss in generality.

The total expected relevance value in the collection is the sum of the total expected relevance value for hits, misses, and near-misses. We refer to this as the *recall-base*. It is stated, as follows,

$$E[\text{Recall-base}, R, A] = E[\text{Hits}, R, A] + E[\text{Misses}, R, A] + E[\text{Near-misses}, R, A] \tag{11}$$

The total expected relevance value of the recall-base can be defined in this way because *Hits* and *Misses* are disjoint; and the expected loss and gain, respectively, of misses and near-misses are complementary. The total expected relevance value of the recall-base represents the maximum relevance value gain that a user could experience given an output and a set of all relevant trees.

In our earlier work [5], we evaluated system performance based on *inferred relevance value*. This basis limits evaluation of tree retrieval effectiveness to precision, because gain is limited to the user seeing relevant nodes *in* the output. ESR goes beyond this, by calculating the user expected gain (or loss) in relevance value based on the user seeing relevant nodes *from* the output as hits, misses and near-misses, and combining these to define recall.

We complete this section with an example calculation of our ESR expected relevance value gain from hits, misses and near-misses, based on the toy collection in Fig. 4a; the navigation model in Table 5; the set of relevance judgments for both *binary relevance* and *relevance by length* of nodes e_3 and e_4 shown in Table 4; and, the three example outputs for System 1 ($R1$), System 2 ($R2$), and System 3 ($R3$), all given in Sect. 4.4.

We begin by illustrating how ESR parameters are calculated for System 1. We recall that ESR relies on three main parameters expressing, respectively, how the user gains relevant information either by the system retrieving the information directly (hits with E[Hits;R, A] in (8)), by the user locating the relevant information via navigation (near-misses with E[Near-misses;R, A] in (10)), or, not at all (misses with $E[Misses;R, A]$ in (9)).

We determine the ESR parameters of expected relevance value of hits, misses and near-misses at rank cut-off $k = 1$ for System 1 in Table 6 (recall that $R1 = e_1$, e_3, e_4). At rank 1, System 1 outputs element e_1. This is not a hit because $A \cap R1_1 = \varnothing$ (6). The misses are $A/R1_1 = e_3$,e_4 (7). The probability that relevant element e_3 can be navigated to from retrieved element e_1 is $p(e_3; R1_1) = 1 - (1 - \tilde{p}(e_3; e_1)) = \tilde{p}(e_3; e_1) = 0.16$ (5). Similarly, the probability that e_4 can be navigated to is $p(e_4; R1_1) = 1 - (1 - \tilde{p}(e_4; e_1)) = 0.11$. Assume binary relevance. The expected relevance value of the near-miss e_3 is $rel(e_3) \times p(e_3;R1_1) = 0.16$ (10). The expected relevance value of the miss e_3 is $rel(e_3) \times (1 - p(e_3;R1_1)) = 0.84$ (9). The expected relevance value of the near-miss e_4 is $rel(e_4) \times p(e_4;R1_1) = 0.11$ (10) and miss e_4 is $rel(e_4) \times (1 - p(e_4;R1_1)) = 0.89$ (9). The recall-base is $0.16 + 0.84 + 0.11 + 0.89 = 2$ (11). The expected relevance value gains, defined by binary relevance, are shown in Row 3 of Table 7. The values in Row 3 in parentheses show the expected relevance value gains if we define the relevance value as relevance by length.

The expected relevance value of hits, misses, and near-misses across rank cut-offs for Systems 1, 2 and 3 are shown for binary relevance and relevance by length (in parentheses) in Table 7. These can be used to understand whether these systems retrieve information directly or whether the user must navigate to find relevant information. For instance, at $k = 3$ (Row 12 to 15), we note that both Systems 1 and 3 retrieve relevant information, whereas in System 2 the system returns near-misses and hence the user will need to spend some effort to locate relevant information.

Table 7 Hits, Near-misses and misses using binary relevance (relevance by length)

1		Hits		Near-misses		Misses	
2	$k = 1$	e_3	e_4	e_3	e_4	e_3	e_4
3	Sys. 1	–	–	0.16(4.8)	0.11(2.2)	0.84(25.2)	0.89(17.8)
4	Sys. 2	–	–	0.16(4.8)	0.11(2.2)	0.84(25.2)	0.89(17.8)
5	Sys. 3	1(30)	–	–	0(0)	–	1(20)
6		Hits		Near-misses		Misses	
7	$k = 2$	e_3	e_4	e_3	e_4	e_3	e_4
8	Sys. 1	0.84(25.2)	–	–	0.11(2.2)	–	0.89(17.8)
9	Sys. 2	–	–	0.16(4.8)	0.23(4.56)	0.84(25.2)	0.77(15.4)
10	Sys. 3	1(30)	–	–	0.11(2.2)	–	0.89(17.8)
11		Hits		Near-misses		Misses	
12	$k = 3$	e_3	e_4	e_3	e_4	e_3	e_4
13	Sys. 1	0.84(25.2)	0.89(17.8)	–	–	–	–
14	Sys. 2	–	–	0.16(4.8)	0.23(4.56)	0.84 (25.2)	0.77(15.4)
15	Sys. 3	1(30)	0.89(17.8)	–	–	–	–

Table 8 Recall-base using binary relevance (relevance by length)

1		Recall-base		
2		$k = 1$	$k = 2$	$k = 3$
3	Sys. 1	2(50)	1.84(45.2)	1.73(43)
4	Sys. 2	2(50)	2(50)	2(50)
5	Sys. 3	2(50)	2(50)	1.89(47.8)

The expected relevance value of the recall-base across rank cut-offs is shown in Table 8. At a given rank cut-off, the size of the recall-base changes inversely to the redundancy in the output up to the given rank, i.e., more redundancy in the output will reduce the size of the recall-base (and the expected relevance value from hits and near-misses). In our example, we note that by using System 3 the user will experience the least redundancy (Row 4 in Table 8) with the greatest gain (from Row 15 in Table 7). Additionally, we note that users of System 2 experience the least overall gain (from Row 14 in Table 7). This corresponds to our earlier assertion that System 2 would have the worst performance and System 3 would have the best.

To summarize, the ESR framework is comprised of four related expected values; namely expected relevance value gain from the user seeing hits in the output (8), expected relevance value loss from (unseen) misses in the collection (9), expected relevance value gain from the user seeing near-misses in the collection (10), and the sum of these three expectations (11), i.e. the recall-base. Next, we show how this framework is used to measure performance for several task-specific approaches in SDR.

6 ESR evaluation measures

In this section, we formulate SDR evaluation measures for SR, HiXEval, XCG, and PRUM (introduced in Sect. 2) within our ESR framework. We consider each by first describing the original measures and then expressing them in ESR. Note that each represents a family of measures, and we formulate only a selection in each. The selected measures are expressed in terms of the expectations defined in the previous section.

6.1 Structural relevance

Structural relevance (SR) [5] is a measure of the user expected gain in relevant information given that the information may be redundant. SR is calculated by summing the expected inferred relevance value gain for the trees in the output:

$$SR(R) = \sum_{i=1}^{k} rel(t_i) \cdot (1 - p(t_i; R_{i-1})) \tag{12}$$

where $rel(t_i)$ is the *inferred* relevance value of subtree t_i, and $p(t_i;R_{i\text{-}1})$ is the probability that the nodes in subtree t_i are seen more than once by the user. In our earlier work [5], SR in Precision (SRP), which is SRP = SR(R)/k, was proposed to measure precision of tree retrieval systems, and to rank systems using mean average precision across rank cut-offs.

To represent SRP in ESR, we replace the expected inferred relevance value gain SR(R) with the expected relevance value gain from hits in ESR, i.e., SR(R) (12) with E[Hits, R, A] (8):

$$\text{ESRP}(R, A) = E[\text{Hits}, R, A]/k. \tag{13}$$

Note that if the inferred relevance value is equal to the assessed relevance value then SR(R) and E[Hits, R, A] are equivalent. For instance, in element retrieval, systems retrieve singletons and the inferred relevance in SR is exactly the judged relevance value in ESR, thus explaining the above equivalence.

A key limitation of inferred relevance value $rel(t_i)$, as originally proposed in SR, is that it is not possible to calculate recall. This is because $rel(t_i)$ can only represent gain from hits or near-misses. Misses (corresponding losses) cannot be accounted for, thus recall cannot be defined. By formulating SR in our framework, as shown next, this limitation is overcome.

Indeed, we recall that the user gains relevant information from hits (8) and near-misses (10). The recall-base in 11 represents the user maximum possible gain. We obtain a measure of recall by dividing the sum of the gain from hits and near-misses by the recall-base. We refer to our recall measure as Structural Relevance in Recall (ESRR):

$$\text{ESRR}(R, A) = \frac{(E[\text{Hits}, \text{R}, \text{A}] + \text{E}[\text{Near} - \text{misses}, \text{R}, \text{A}])}{E[\text{Recall-base}, R, A]} \tag{14}$$

In the case where a user cannot navigate ($\tilde{p}(t_i; t_j) = 0$ for all trees in the collection) and assuming binary relevance ($rel(a) = 1$ for relevant trees), it can be shown that ESRP and ESRR reduce to classical precision (r/k) and recall (r/N), respectively.

6.2 Highlighting XML retrieval evaluation

Highlighting XML evaluation (HiXEval) proposed in Pehcevski and Thom [31], and further finalized in Kamps et al. [22], was developed to evaluate the performance of systems that retrieve (or can be modelled as retrieving) passages, where a passage is a block of text, delineated or not with XML tags.

HiXEval exploits the relevance assessment methodology used at INEX since 2005 [36], where human judges highlight the relevant passages in retrieved (pooled) documents. With this methodology, for a given information need, the relevant parts in documents are those that have been highlighted by the human judges [36]. HiXEval measures precision and recall based on the amount of relevant information retrieved; the amount of relevant information in the collection; and the overlap of the relevant text in retrieved passages. The "amount of information" is measured using the character length of passages.

In HiXEval, for a given information need the total relevance value of the information contained across all documents in the collection is given by the number of highlighted characters in the whole collection. Let T_{rel} denote the number of characters in the relevant (highlighted) text in the collection. The relevance value of a retrieved passage in HiXEval is the character length of the relevant text in the passage. If the relevant text overlaps with another retrieved passage, then the overlapped text is relevant to the user with probability $\alpha \in [0, 1]$, where α refers to the user tolerance to overlap. HiXEval assumes that user navigation does not extend beyond the boundaries of retrieved passages. Thus, HiXEval considers redundancy as only occurring between adjacent retrieved text passages overlapping each other.

For a retrieved passage e, the user gain in relevant information is given by $rsize(e)$, which is defined as follows. Let $size(e)$ denote the size of the retrieved passage. Let $rel(e)$ denote the size of the relevant text in the passage. Let $rov(e)$ denote the number of characters in the relevant text that is overlapped with a higher-ranked passage in the output. The gain is stated as follows:

$$rsize(e) = rel(e) - (1 - \alpha) \times rov(e) \tag{15}$$

Based on the above, numerous measures can be obtained for measuring precision and recall in passage retrieval. In this section, we consider two HiXEval measures; namely interpolated precision (iP) and interpolated recall (iR)[7]. Interpolated precision is the user gain in relevant information divided by the number of characters retrieved (16). Interpolated recall is the user gain in relevant information divided by the total relevance value in the collection (17).

$$iP@r = \frac{\sum_{i=1}^{r} rsize(e_i)}{\sum_{i=1}^{r} size(e_i)} \tag{16}$$

$$iR@r = \frac{\sum_{i=1}^{r} rsize(e_i)}{T_{rel}} \tag{17}$$

where $R = e_1, e_2, \ldots, e_k$ is a ranked list of k passages and $r \in [1, k]$ is a rank. Mean average precision across either rank cut-offs or recall points is used to rank systems.

[7] At INEX, generalized precision (gP) and generalized recall (gR) are currently the official (HiXEval-based) measures. Interpolated precision (iP) and interpolated recall (iR) have been used in the past as official measures at INEX [22]. In Sect. 7, we validate ESR for iP, iR, and gP.

We formulate now HiXEval, i.e. iP and iR, in the ESR framework. First, we define the relevance value $rel(a)$ as the number of characters in the relevant text in the nodes of the tree a. Second, we note that overlap in tree retrieval is a specific case of redundancy where trees in the output share nodes in common, and that this can be accounted for in ESR using an appropriate user navigation model. Third, let T_{rel} be the number of characters in the relevant text in the collection and $size(t)$ denote the number of characters in the nodes of tree t.

We replace the user gain in relevant information $rsize()$ (15) with the sum of the gain from hits (8). We limit gain to hits because HiXEval (and XCG) measures limit consideration of user navigation to within retrieved elements. This is fully accounted for in ESR with hits. Indeed, as stated in Sect. 4.1, near-misses in HiXEval (and XCG) are defined differently than in ESR[8]. We obtain the following ESR measures for interpolated precision (SRiP) and recall (SRiR), stated without derivation,

$$\mathrm{SRiP}(R,A) = \frac{E[\mathrm{Hits,R,A}]}{\sum_{i=1}^{k} size(t_i)} \tag{18}$$

$$\mathrm{SRiR}(R,A) = \frac{E[\mathrm{Hits,R,A}]}{T_{rel}} \tag{19}$$

The key differences between iP/iR and SRiP/SRiR are that the latter are based on tree retrieval and consider a broader notion of redundancy than overlap (which is a special case of redundancy). The SRiP/SRiR measures above can be applied to any search task that can be modelled using tree retrieval. This demonstrates an important advantage when using ESR in that an evaluation approach like HiXEval can be applied to tasks that go beyond the search paradigm, here passage retrieval, for which it was originally proposed.

6.3 Extended cumulated gain

Extended cumulated gain (XCG) [24] is a family of measures that evaluate the user gain in relevant information from an actual system compared to the gain possible from an ideal system (see Sect. 2 for details on ideality). One of the XCG measures is the normalized extended cumulated gain (NXCG), which we formulate now within our ESR framework.

NXCG is the ratio of the user cumulated gain in relevant information from an actual system compared to the cumulated gain from an ideal system. The cumulated gain xCG[k] (20) is the user gain after consulting k ranks from the actual system. The ideal cumulated gain xCI[k] (shown in (21)) is the user gain after consulting k ranks from the ideal system. NXCG is defined as their ratio (22).

$$\mathrm{xCG}[k] = \sum_{i=1}^{k} \mathrm{xG}[i] \tag{20}$$

$$\mathrm{xCI}[k] = \sum_{i=1}^{k} \mathrm{xI}[i] \tag{21}$$

[8] Near-misses are defined for XCG in Kazai and Lalmas [24] as retrieved sub-documents that, may or may not be relevant, but which can be navigated from by the user to see non-retrieved, relevant information. Whereas, in ESR, we reverse this definition.

$$\text{NXCG}[k] = \frac{\text{xCG[k]}}{\text{xCI}[k]} \tag{22}$$

where $\text{xG}[i]$ is the gain from the ith element in the actual system output $R = e_1, e_2, \ldots, e_k$, and $\text{xI}[i]$ is the gain from the ith element in the ideal system output $I = \text{ideal}_1, \text{ideal}_2, \ldots, \text{ideal}_n$. XCG has been developed for measuring element retrieval systems[9], and thus e_i is an XML element in the output, and ideal_i is an element in the set of assessed ideal elements. Averaged NXCG at a given rank cut-off is used to rank systems.

Relevance in XCG is considered as follows. At each rank consulted, the user gains relevant information depending on whether the consulted element contains relevant text and whether its text overlaps with other retrieved elements. There are numerous ways in XCG to calculate the user gain in relevant information, depending on how the relevance of elements has been determined (which has changed over the years at INEX [36]). For illustrative purposes, we use the same approach described in Sect. 6.2, which is based on the amount of highlighted characters.

Let $size(e)$ denote the number of characters in a retrieved element. Let $rsize(e)$ denote the number of characters in the text of a retrieved element that are relevant to the user. The calculation of $rsize(e)$ is shown in Sect. 6.2 in (15). The actual gain in (20) is then $\text{xG}[i] = rsize(e_i)/size(e_i)$. The ideal gain in (21) is then $\text{xI}[i] = rsize(\text{ideal}_i)/size(\text{ideal}_i)$.

As for HiXEval, we limit gain in XCG to hits. We now formulate NXCG using ESR. For this, we first consider xCG and xCI within our ESR framework. We start with xCG (20). The user total expected relevance value gain is the sum of hits (8). We refer to this as the cumulated gain CG and show this below in (23).

We now discuss xCI (21). In this work, we propose an alternative that mitigates the instability caused by ideality cited in Kazai et al. [26]. We propose that ideal cumulated gain be replaced with a *desired* cumulated gain. The latter (desired) refers to the cumulated gain that a user expects by spending a given effort. Similar approaches to measuring effort-gain relationships can be found in expected search length [13] and PRecall [37]. The desired cumulated gain can be calculated as follows. Let m denote the desired effort spent (by number of ranks) to satisfy the user information need. Let l denote the desired recall to satisfy the user information need. Let i denote a rank cut-off. The total relevance value needed to satisfy the user information need is the recall-base (11) times the desired recall. If we divide this by the desired effort, then we obtain the desired gain per rank. We multiply this by the current rank cut-off to get the cumulated desired gain. This is shown in (24).

Thus, we can now express NXCG within ESR. We divide the user gain from hits $\text{CG}[i]$ by the desired cumulated gain $\text{CD}[i]$. We call this measure normalized extended cumulated gain in ESR (NSRCG), shown in (25).

$$\text{CG}[i] = E[\text{Hits}, R_i, A] \tag{23}$$

$$\text{CD}[i] = i \times l \times E[\text{Recall-base}, R_i, A]/m \tag{24}$$

$$\text{NSRCG}[i] = \frac{\text{CG[i]}}{\text{CD}[i]} \tag{25}$$

where $i \in [1, k]$ is the number of ranks consulted, $l \in (0, 1]$ is the desired recall, and m is the desired effort.

NSRCG does not require an ideality assumption, which allows it (and other measures in the XCG family) to be applied to any SDR search task that can be modelled as tree retrieval.

[9] Although this does mean that XCG cannot be extended to evaluate passage retrieval.

6.4 Precision-recall with user modeling (PRUM)

Precision-Recall with User Modelling (PRUM) [35] is a measure of whether a user sees a desired number of ideal elements. It is the ratio of the expected number of rank positions where the user gains relevant information by seeing ideal elements compared to the expected number of rank positions that the user consults to satisfy their information need. It is defined, given an information need, desired recall, and output, as follows,

$$\mathrm{PRUM} = \frac{E[\#\text{of rank positions user sees ideal}]}{E[\#\text{ of rank positions consulted}]} \tag{26}$$

where average PRUM at the user desired recall-level is used to rank systems.

To see elements, a user either *consults* the system output or *navigates* from a retrieved element. PRUM is calculated by enumerating all of the possible *scenarios* of consultations and navigations that result in the user seeing the desired number of ideal elements. Let i denote the desired number of ideal elements. Each scenario includes the number of ranks consulted $C(i)$ and the number of ranks where the user gains relevant information by seeing ideal elements $\mathrm{CL}(i)$. The probability $P(S)$ of each scenario S occurring can be calculated based on the taken (and not taken) navigations. The expected ranks are calculated by conditioning $C(i)$ and $\mathrm{CL}(i)$, respectively, on the probability $P(S)$ such that PRUM $= E[\mathrm{CL}(i)]/E[C(i)]$.

We illustrate with an example. Consider calculating PRUM for a system that outputs $R = e_3, e_1, e_4$ for the document shown in Fig. 4a. The calculation of PRUM is as follows, and summarized in Table 9. Let the ideal elements be e_3 and e_4, which are outputs in rank positions 1 and 3, respectively. Let the desired recall-level be two ideal elements $i = 2$. Assume that the only possible navigation is from element e_1 to element e_4. Given this navigation model, there are two possible scenarios:

(A) the user sees e_3 by consulting the ranked list and sees e_4 by navigating from e_1, so the user consults the ranked list two times ($C(2) = 2$) and there are two ranks (1 and 2) that lead to seeing unique, ideal elements ($\mathrm{CL}(2) = 2$);

(B) the user sees e_3 and e_4 by consulting the ranked list and does not navigate to e_4 from e_1, so the user consults the ranked list three times ($C(2) = 3$) and there are two ranks (1 and 3) that lead to seeing unique, ideal elements ($\mathrm{CL}(2) = 2$);

Table 9 Example of PRUM ($P(e_1 \rightsquigarrow e_4) = 0.2$)

	Scenarios			
Rank	A	B		
1 (e_3)	e_3	e_3		
2 (e_1)	e_4			
3 (e_4)		e_4		
P(S)	$P(e_1 \rightsquigarrow e_4)$ 0.2	$1 - P(e_1 \rightsquigarrow e_4)$ 0.8	EXP	PRUM
CL(2)	2	2	2	
C(2)	2	3	2.8	0.714

The probability of each scenario occurring is determined as follows. Let $P(f \rightsquigarrow e)$ denote user navigation as the probability[10] that the user has seen element e given that the user has seen element f. Assume user navigation is $P(e_1 \rightsquigarrow e_4) = 0.2$. For scenario A, the user navigates to e_4 from e_1, i.e., $P(A) = P(e_1 \rightsquigarrow e_4)$. For scenario B, the user does not navigate to e_4 from e_1, i.e., $P(B) = 1 - P(e_1 \rightsquigarrow e_4)$. Table 9 summarizes the scenarios A and B, where the row $P(S)$ shows the probability of each scenario occurring, the column EXP shows the expected values of CL(2) and C(2), and the column PRUM shows that the PRUM precision for this example is 0.714.

Next, we use ESR to formulate PRUM. For this, we need to express CL and C within ESR. The user desire is to see non-redundant, relevant information at each rank position consulted. Note that PRUM, as defined in Piwowarski et al. [35], does not consider graded assessments. Thus, let us assume binary relevance values. User navigation in ESR and PRUM is similar (they differ in their probabilistic interpretation, but, in general, both consider navigation between pairs of nodes). Thus, unlike HiXEval and XCG, PRUM considers near-misses in ESR and the user gain in PRUM is the sum of hits (8) and near-misses (10). The desired number of consultations of the output is equal to the gain because each relevant tree contributes up to 1 to the gain. Thus, the desired number of ranks is stated as:

$$\mathrm{CL}(i) = E[\text{Hits}, R_i, A] + E[\text{Near-misses}, R_i, A] \tag{27}$$

where $i \in [1, k]$ is a rank cut-off, and $rel(a) = 1$ for relevant trees $a \in A$, and $rel(a) = 0$ otherwise.

The number of ranks that the user consults to satisfy a given information need is obtained by calculating the rank cut-off for a given recall level. Let r be the user desired recall level. Let m be the minimum rank cut-off where the user desired recall level is achieved. This cut-off is calculated using ESRR(R, A) (14) by evaluating ESRR across rank cut-offs $m \in [1..k]$ until ESRR(R_m, A) is greater than or equal to the desired recall r:

$$C = m, \text{ where } \mathrm{ESRR}(R_m, A) \geq r \tag{28}$$

Precision in PRUM using ESR (SRPRUM) is the ratio between the desired number of ranks to achieve a given recall-level CL(C) and the rank cut-off C where a given recall-level is achieved, which is

$$\mathrm{SRPRUM} = \mathrm{CL}(C)/C \tag{29}$$

where CL(C) (27) is the desired number of consultations of the output to achieve recall r, and C (28) is the actual number of consultations to achieve recall r.

ESR does not rely on ideality, as does not SRPRUM. Similarly, assuming binary relevance of judged trees, SRPRUM can be applied to any (SDR and beyond) search task that can be modelled as tree retrieval.

[10] $P(f \rightsquigarrow e)$ in PRUM is different from $\tilde{p}(e;f)$ in ESR. $P(e \rightsquigarrow f)$ is the probability that a user who has seen element f has also seen element e. Whereas, $\tilde{p}(e;f)$ is the probability that a user who navigates from element f will navigate to element e. The difference lies in how user navigation is assessed in e.g. user studies. $P(f \rightsquigarrow e)$ is determined by asking the reader post-assessment whether specific ideal elements were seen or not. In contrast, $\tilde{p}(e;f)$ is determined by tracking the reader's attention and assuming that navigation is independent of relevance.

6.5 Calculating SR, HiXEval, XCG and PRUM using ESR

Let us now continue our illustrative example of ESR evaluation. In Sect. 4.4, we introduced a toy collection (Fig. 4a), a navigation model for the collection (Table 5), a set of relevance judgments for both *binary relevance* and *relevance by length* of nodes e_3 and e_4 (Table 4), and three example outputs $R1$, $R2$, and $R3$ (Table 6) where System 3 ($R3$) is the best system, System 1 ($R1$) is the second-best system, and System 2 ($R2$) is the worst system. In Sect. 5, we calculated the ESR expected relevance value gain from hits, misses and near-misses across the rank positions of each example system output (as summarized in Table 8). In this section, we demonstrate how we use our ESR expectations to calculate the ESR measures proposed in this section.

Let us begin our demonstration by calculating SR, HiXEval, XCG and PRUM for System 3 ($R3 = e_3$, e_1, e_4) at rank cut-off $k = 2$. We begin by calculating structural relevance in precision (ESRP) and structural relevance in recall (ESRR). Assume binary relevance. The expected relevance values for $R3_2$ are found in Row 10 of Table 7. The sum of the hits in Row 10 is E[Hits, $R3_2$, A] = 1 + 0 = 1. The sum of the near-misses in Row 10 is E[Near-misses, $R3_2$, A] = 0 + 0.11 = 0.11. The sum of the misses in Row 10 is E[Misses, $R3_2$, A] = 0 + 0.89 = 0.89. From Row 5 of Table 8, our recall-base is E[Recall-base, $R3_2$, A] = 2. Precision is ESRP($R3$, A)@2 = E[Hits, $R3_2$, A]/2 = 0.5 (13) and the recall ESRR($R3$,A)@2 = (1 + 0.11)/2 = 0.555 (14). Table 11 tabulates ESRP and ESRR across rank cut-offs for all systems.

Next, we calculate interpolated precision (SRiP) and recall (SRiR) in HiXEval. Assume relevance by length. The expected relevance values for $R3_2$ can be found in Row 10 of Table 7. The sum of the hits E[Hits, $R3_2$, A] in Row 10 is 30 + 0 = 30. The recall-base is E[Recall-base, $R3_2$, A] = 50 from Row 5 of Table 8. Interpolated precision is SRiP@2 = 30/130 = 0.231 (18). Interpolated recall is SRiR@2 = 30/50 = 0.6 (19). Table 12 tabulates SRiP and SRiR across rank cut-offs for all systems.

Next, we use normalized cumulated gain in ESR (NSRCG) to calculate XCG. Again, assume relevance by length. The expected relevance values for $R3_2$ can be found in Row 10 of Table 7 and the recall-base in Row 5 of Table 8. Let the desired recall be l = 100%

Table 10 Summary of SDR measures

INEX	ESR
$\mathrm{SRP}(R) = \frac{\sum_{i=1}^{k} rel(t_i)\cdot(1-p(t_i;R_{i-1}))}{k}$	$\mathrm{ESRP} = \frac{E[\mathrm{Hits,R,A}]}{k}$
–	$\mathrm{ESRR} = \frac{E[\mathrm{Hits \wedge Near\text{-}misses,R,A}]}{E[\mathrm{Recall\text{-}base,R,A}]}$
$iP@r = \frac{\sum_{i=1}^{r} rsize(e_i)}{\sum_{i=1}^{r} size(e_i)}$	$\mathrm{SRiP} = \frac{E[\mathrm{Hits,R,A}]}{\sum_{i=1}^{k} size(t_i)}$
$iR@r = \frac{\sum_{i=1}^{r} rsize(e_i)}{T_{rel}}$	$\mathrm{SRiR} = \frac{E[\mathrm{Hits,R,A}]}{T_{rel}}$
$\mathrm{xCG}[k] = \sum_{i=1}^{k}\mathrm{xG}[i]$	$\mathrm{CG}[i] = E[\mathrm{Hits}, R_i, A]$
$\mathrm{xCI}[k] = \sum_{i=1}^{k}\mathrm{xI}[i]$	$\mathrm{CD}[i] = i \times l \times E[\mathrm{Recall\text{-}base}, R_i, A]/m$
$\mathrm{NXCG}[k] = \mathrm{xCG}[k]/\mathrm{xCI}[k]$	$\mathrm{NSRCG}[i] = \mathrm{CG}[i]/\mathrm{CD}[i]$
$\mathrm{PRUM} = \frac{E[\#\text{of rank pos. user sees ideal}]}{E[\#\text{of rank pos. consulted}]}$	$\mathrm{CL}(i) = E[\mathrm{Hits} \wedge \mathrm{Near\text{-}misses}, R_i, A]$
	$C = m$, where $\mathrm{ESRR}(R_m, A) \geq r$
	$\mathrm{SRPRUM} = \mathrm{CL}(C)/C$

Table 11 Structural relevance in precision (ESRP) and recall (ESRR)

	ESRP@1(ESRR@1)	ESRP@2(ESRR@2)	ESRP@3(ESRR@3)
System 1	0 (0.135)	0.42 (0.516)	0.577 (1)
System 2	0 (0.135)	0 (0.194)	0 (0.194)
System 3	1 (0.5)	0.5 (0.555)	0.63 (1)

Table 12 Precision and recall for HiXEval using ESR

	SRiP@1(SRiR@1)	SRiP@2(SRiR@2)	SRiP@3(SRiR@3)
System 1	0 (0)	0.194 (0.558)	0.287 (1)
System 2	0 (0)	0 (0)	0 (0)
System 3	1 (0.6)	0.231 (0.6)	0.319 (1)

Table 13 XCG using ESR

	CG[k]/CD[k] = NSRCG[k]		
	$k = 1$	$k = 2$	$k = 3$
System 1	0/25 = 0	25.2/45.2 = 0.56	25.2/64.5 = 0.39
System 2	0/25 = 0	0/50 = 0	0/75 = 0
System 3	30/25 = 1.2	30/50 = 0.6	30/71.7 = 0.42

Table 14 PRUM using ESR

	System 1	System 2	System 3
SRPRUM	0.577	0.129	0.63

and let the desired effort be $m = 2$ ranks. The desired cumulated gain is CD[2] = 2 × 100% × 50/2 = 50 (24). Similarly, the user's expected gain is CG[2] = 30 (23). Normalized cumulated gain is NSRCG[2] = CG[2]/CD[2] = 30/50 = 0.6 (25). Table 13 tabulates CD, CG, and NSRCG across rank cut-offs for all systems.

Finally, we calculate PRUM using precision-recall with user modelling in ESR (SRPRUM). Assume binary relevance values. The expected relevance values for $R3_2$ can be found in Row 10 of Table 7 and the recall-base in Row 5 of Table 8. Let the required number of ranks to achieve the desired recall be $C = 2$ (28), which corresponds to a desired recall of $l \geq 0.555$ using ESRR in Table 11. The expected number of rank positions where the user gains relevant information at $C = 2$ is CL(2) = 1 + 0.11 = 1.11 (27). Precision-recall with user modelling in ESR is SRPRUM =1.11/2 = 0.555 (29). Table 14 tabulates SRPRUM at a desired recall of $l = 100\%$ for all systems.

Tables 11, 12, 13, and 14 show our results for ESRP/ESRR, SRiP/SRiR, NSRCG, and SRPRUM, respectively, for all systems. We can observe the following. System 2 does not retrieve hits. At rank cut-off 3 (which is the maximum recall for all systems), for SR, HiXEval and PRUM, the system ranking is $R3 \succ R1 \succ R2$, where $\succ$ denotes the left-hand system performing better than the right-hand system. Using XCG, System 1 and 3 are tied and the system ranking is $(R3, R1) \succ R2$. As expected, in Sect. 4.4, both Systems 1 and 3 outperformed System 2 for all measures. Similarly, in terms of precision, System 3 outperforms System 1 and is the best system. Using XCG, System 3 outperforms System 1. Overall, we obtain the expected ranking of systems as predicted above in Sect. 4.4.

6.6 Discussion

In this section, we have demonstrated how current SDR measures can be formulated and calculated in ESR. Table 10 summarizes the original measures (as proposed at INEX) and the corresponding ESR measures. When formulated within ESR, the resulting measures are not necessarily exact equivalents of the original measures. We have however shown in our previous work on SR in [5–7], upon which ESR is based, that the probabilistic approach presented here for measuring precision is a reliable performance measure with respect to both XCG and HiXEval. The goal of this work is to provide a framework in which new measures for SDR evaluation can be developed. Our intention in this section was to show how current SDR measures could have been (directly) expressed within our ESR framework. Our future work will be to further refine our proposals if needed (e.g. accounting for near-misses and overlap in INEX measures), and then to fully validate the ESR framework across these and additional measures, and search tasks.

The benefit of ESR is that we have now SDR measures that become inherently comparable because they rely on the same set of parameters, namely E[Hits;R, A], E[Misses;R, A], E[Near-misses;R, A], and E[Recall-base;R, A] (8, 9, 10, and 11, respectively). In addition, ESR provides a convenient way for measures to share common models of relevance value and user navigation; provides a means to apply evaluation approaches developed for different paradigms to tree retrieval; and, system performance can be compared in both general terms of how users gain relevant information (via hits, misses and near-misses) and how a system fulfills a specific search task (via task-specific measures). We believe that the flexibility to support such varied measures in a single framework is an important advancement for the development and evaluation of complex search tasks, many of which are to come in the near future.

7 System rankings using ESR measures

In this section, we compare our ESR measures, to their originals by evaluating three SDR tasks, namely, Focused (2006, 2007), Best In Context (2006), and Relevant In Context (2007), all carried out in INEX. To calculate user navigation probabilities, we use the depth-weighted summary model of navigation described in Sect. 4.2 (originally proposed and validated in Ali et al. [7]). For each ESR measure tested, we compared the system rankings from ESR to the official INEX results using Kendall's Tau, a common way to compare rankings of systems in information retrieval evaluation. Kendall's Tau (τ) indicates whether two separate rankings, as generated in our case by two evaluation measures (an ESR measure and its INEX original counterpart), are positively ($\tau > 0$) or negatively ($\tau < 0$) ordered. The p value is the probability that the compared rankings are not correlated. If a p value is <0.05 then the two measures are correlated in terms of how they order systems. So, in comparing system rankings for a current measure versus a reference measure, the rankings will be either positively correlated, negatively correlated, or not correlated. A positive correlation implies that our ESR measure is an appropriate representation of its original INEX counterpart.

To calculate recall-points for ESR measures SRiP (18) and SRiP2 (30), we selected SRiR2 (31). To calculate recall-points for ESR measure ESRP (13), we selected ESRR (14).

We noted in Sect. 5 that HiXEval and XCG, like SRP, do not consider near-misses. For comparative purposes, we propose ESR formulations of HiXEval and XCG where near-misses are included. Interpolated precision and recall in HiXEval including near-misses (SRiP2 and SRiR2, respectively) are:

$$\mathrm{SRiP2}(R,A) = \frac{E[\mathrm{Hits, R, A}] + \mathrm{E}[\mathrm{Near - Misses, R, A}]}{\sum_{i=1}^{k} size(t_i)} \tag{30}$$

$$\mathrm{SRiR2}(R,A) = \frac{E[\mathrm{Hits, R, A}] + \mathrm{E}[\mathrm{Near - Misses, R, A}]}{T_{rel}} \tag{31}$$

Similarly, normalized extended cumulated gain (NSRCG2) is:

$$\mathrm{CG}[i] = E[\mathrm{Hits}, R_i, A] + E[\text{Near-Misses}, R, A] \tag{32}$$

$$\mathrm{CD}[i] = i \times l \times E[\text{Recall-base}, R_i, A]/m \tag{33}$$

$$\mathrm{NSRCG2}[i] = \frac{\mathrm{CG[i]}}{\mathrm{CD}[i]} \tag{34}$$

where $i \in [1, k]$ is the number of ranks consulted, $l \in (0, 1]$ is the desired recall, and m is the desired effort. By including near-misses, our updated ESR formulations of XCG and HiXEval capture user navigation, which was not the case with our initial formulations, i.e. nXCG and iP/iR. In this work, recall points for ESR HiXEval measures are calculated using SRiR2 (31). Finally, mean-average precision is calculated over 101 recall points.

7.1 Focused task (2006): nXCG and iP

In the *Focused Task*, a system is tasked with retrieving non-overlapping, focused document parts. The reported INEX measures for this task in 2006 are MAiP (16 mean-averaged) at rank cut-off $k = 1000$, and nXCG (22) at rank cut-off $k = 10$. For nXCG, the reported INEX measures are further sub-divided into nXCG ON (the user does not tolerate overlapped text) and nXCG OFF (the user tolerates overlapped text). From Sect. 6.2, overlap ON means $\alpha = 1$ and overlap OFF $\alpha = 0$. MAiP is reported with overlap ON.

We evaluated 43 runs across 107 topics using the ESR measures NSRCG (25) at rank cut-off $k = 10$, NSRCG2 (34) at rank cut-off $k = 10$, MASRiP (18) at rank cut-off $k = 1000$, and MASRiP2 (30) at rank cut-off $k = 1000$. The 43 systems included the top-30 officially best systems (as determined using nXCG ON) and 13 randomly selected systems.

Tables 15 and 16 show the system ranking comparison results between the original (INEX) and ESR measures using Kendall's Tau and p value (in parentheses). Table 15 shows that the system rankings from NSRCG are negatively ordered ($\tau < 0$) with nXCG OFF (where user tolerates overlaps). With overlap ON, system rankings via NSRCG are not correlated (p value < 0). System rankings via NSRCG2 are positively ordered ($\tau > 0$) with nXCG OFF but not correlated (p value < 0.05). However, with overlap ON, NSRCG2 is positively ordered ($\tau > 0$) with nXCG and correlated (p value < 0.05). Table 16 shows that the system rankings via MASRiP and MASRiP2 are positively ordered ($\tau > 0$) and positively correlated ($\tau > 0$, p value < 0.05) with those via MAiP.

Thus, in the focused task, NSRCG and NSRCG2 are not appropriate representations of nXCG OFF. But, NSRCG2 is with respect to nXCG ON, whereas NSRCG is not. In

Table 15 INEX 2006 Focused Task, XCG, 107 topics, 43 Systems

	NSRCG[10]	NSRCG2[10]
nXCG[10] OFF	−0.25(0.02)	0.01(0.92)
nXCG[10] ON	−0.072(0.5)	0.26 (0.01)

Table 16 INEX 2006 Focused Task, MAIP, 107 topics, 43 Systems

	MASRiP	MASRiP2
MAIP	0.30(0.00)	0.58(0.00)

addition, both MASRiP and MASRiP2 are both appropriate representations of iP with overlap ON. Based on the negative ordering of ESR measures for nXCG with overlap OFF, we theorize that ESR, as defined thus far, does not capture users who tolerate seeing overlapped information. This is likely because we have limited our consideration of user navigation, in this work, to between nodes and not within the same node.

7.2 Focused task (2007): iP

This task is the same as for 2006 above. However, the official measure for this task is iP (16) at the recall point 0.01 calculated using iR (17) with overlap ON, i.e. $\alpha = 1$. We evaluated 77 runs across 102 topics using the ESR measures MASRiP (18) at rank cut-off $k = 1000$; SRiP (18) at recall point 0.01; MASRiP (30) at rank cut-off $k = 1000$; and, SRiP2 (30) at recall point 0.01.

Table 17 shows the system ranking comparison results between the original (INEX) and ESR measures. MASRiP, MASRiP2, SRiP, and SRiP2 are positively ordered ($\tau > 0$) and positively correlated ($\tau > 0$, p value <0.05) to iP. The mean-averaged ESR measures (MASRiP and MASRiP2) have better rank correlation (τ is higher) than their corresponding rank cut-off measures (SRiP and SRiP2, respectively). These results agree with our results in Sect. 7.1 that SRiP and SRiP2 are appropriate representations of iP for the focused task.

7.3 Best in context task (2006): EPRUM

In this search task, a system is asked to retrieve the single, most focused, relevant part of a document. The official INEX measure for this task is EPRUM[11] [33] (introduced in Sect. 2) which is a simplified version of PRUM. Navigation in EPRUM uses a proximity measure based on a scalar parameter A, representing the distance, in the document, that a user will navigate from a given entry point to locate relevant information. $A = 0.1$ refers to a user willing to navigate to information very close to the entry point, whereas $A = 100$

[11] EPRUM models the expected relative effort a user spends to achieve a desired recall l using an actual system versus the effort using an ideal system. It is calculated as follows: EPRUM$(l) = \mathbb{E}(\frac{min_I(l)}{min_s(l)})$ where $min_I(l)$ is the minimum number of consulted elements to achieve a recall l in an ideal output, and $min_s(l)$ is the minimum number of consulted elements over all possible scenarios to achieve a recall l in the actual output.

Table 17 INEX 2007 focused task, 102 topics, 77 systems

	SRiP_0.01	MASRiP	SRiP2_0.01	MASRiP2
iP_0.01	0.17(0.03)	0.54(0.00)	0.38(0.00)	0.53(0.00)

refers to a user willing to navigate to information much further away to the entry point. The reported INEX measures included $A = 0.1, 1, 10, 100$.

We evaluated 64 runs across 107 topics using the ESR measures SRPRUM (29) and ESRP (13 mean-averaged across recall points). We used ESRR (14) for calculating recall points. Table 18 shows the system ranking comparison results between the original (INEX) and ESR measures. The system rankings from SRPRUM are positively ordered ($\tau > 0$) and positively correlated ($\tau > 0$, p value <0.05) to EPRUM for all values of A. The system rankings from ESRP are positively ordered ($\tau > 0$) but uncorrelated (p value < 0.05) to EPRUM for all values of A. We conclude that SRPRUM is an appropriate representation of EPRUM for the best in context task.

7.4 Relevant in context task (2007): MAgP

In the this search task, the system is tasked with retrieving focused answers grouped per document. The official measure for this task is MAgP, which is *generalized precision* mean-averaged across recall points where recall is measured using *generalized recall* (*gR*) [21]. In HiXEval, to account for near-misses, generalized measures have been proposed. These measures extend interpolated measures by accounting for user gain at the document-level. For instance, if a system retrieves a passage with x number of relevant characters from a document containing y relevant characters, then, depending on how navigation is modelled, the user is modelled to see between x and y relevant characters from the given document. This is akin to how, in this work, navigation $p(t_i;t_j)$ (in (4)) is only non-zero between sub-documents in the same document. In this way, navigation in generalized measures goes beyond retrieved text passages and is more akin to near-misses as defined in ESR. This approach solves the navigational limitations mentioned in Sect. 6.2 in the case where documents in the collection can be considered as single passages (such as Wikipedia articles). But, it remains to be seen whether this approach can address cases where documents cannot be considered as single passages such as in online books or semantically linked data.

We evaluated 77 across 102 topics using the ESR measures MASRiP (18) at rank cut-off $k = 1000$ and MASRiP2 (30) at rank cut-off $k = 1000$. Table 19 shows the system ranking comparison results between the original (INEX) and ESR measures. Both MASRiP and MASRiP2 are positively ordered ($\tau > 0$) and positively correlated ($\tau > 0$, p value

Table 18 INEX 2006 best in context task, 107 topics, 64 systems

	SRPRUM	ESRP
EPRUM A= 0.01	0.32(0.00)	0.02(0.83)
EPRUM A= 1	0.66(0.00)	0.10(0.2)
EPRUM A = 10	0.60(0.00)	0.12(0.15)
EPRUM A=100	0.58(0.00)	0.09(0.25)

Table 19 INEX 2007 relevant in context task, 102 topics, 77 systems

	MASRiP	MASRiP2
MAgP	0.34(0.00)	0.34(0.00)

Table 20 Final results (Yes: $\tau > 0.25$, No: otherwise)

Task	INEX	ESR	Appropriate
Focused	nXCG, overlap OFF	nSRCG	No
Focused	nXCG, overlap OFF	nSRCG2	No
Focused	nXCG, overlap ON	nSRCG	No
Focused	nXCG, overlap ON	nSRCG2	Yes
Focused	MAiP, overlap ON	MASRiP	Yes
Focused	MAiP, overlap ON	MASRiP2	Yes
Focused	iP, overlap ON, iR = 0.01	MASRiP	Yes
Focused	iP, overlap ON, iR = 0.01	MASRiP2	Yes
Focused	iP, overlap ON, iR = 0.01	SRiP	No
Focused	iP, overlap ON, iR = 0.01	SRiP2	Yes
Best in context	EPRUM	SRPRUM	Yes
Best in context	EPRUM	ESRP	No
Relevant in context	MAgP	MASRiP	Yes
Relevant in context	MAgP	MASRiP2	Yes

<0.05) to MAgP. We conclude that MASRiP and MASRiP2 appropriate representations of MAgP for the relevant in context task.

7.5 Summary

In this section, we tested our ESR proposals for XCG, (E)PRUM and HiXEval to their counterpart INEX measures. We summarize our results in Table 20.

For XCG, (E)PRUM, and HiXEval, appropriate representations in ESR measures exist for all tasks except in the focused task where the user tolerates overlap (i.e., overlap OFF). In INEX, overlap is evaluated using a factor on the gain to represent the user's tolerance for seeing relevant, retrieved information more than once [44]. With how we have defined ESR, gain is not explicitly penalized in cases of overlapped results. Instead, we relied on the notion of redundancy to account for it. We theorize that the navigation model used in this study does not adequately address the issue of overlap, as considered at INEX. Indeed, a better approach would be to sub-divide, into overlapped and non-overlapped relevant text, the ESR partition for relevant text in the collection that has been retrieved, i.e., $E[\text{Hits}, R, A]$ in (8). This would allow us to introduce an overlap parameter α akin to the one used for HiXEval and XCG. We leave this for future work.

Our results however demonstrate the advantages of our ESR common basis of performance based on hits, misses, near-misses using a model based on relevance, navigation and redundancy. ESR allows us to generalize performance calculation across approaches (e.g., HiXEval, XCG, and PRUM). The contrast in measures across tasks allows us to isolate problems in evaluation (e.g., overlap). In ESR, we can address these problems by refining our common basis.

8 Conclusions and future work

In this paper, we proposed a general framework, called Extended Structural Relevance (ESR), in which to express evaluation measures for SDR. This paper follows from our previous work [5] on evaluating tree retrieval, which many of the current search tasks in SDR, are special cases. In this previous work, we identified three main pillars for evaluating the performance of SDR systems, namely, relevance, navigation and redundancy. ESR incorporates relevance, navigation and redundancy into a single probabilistic framework, and thus allows us to calculate the user expected gain in relevant information accounting for hits, misses or near-misses. We use these expectations as parameters defining a basis to formulate evaluation measures for SDR.

Our aim was to overcome a main drawback that arose from the development of task-specific measures in SDR, i.e., current SDR measures of performance cannot easily be compared with respect to each other and across search tasks. Our experimental results validated that task-specific measures at INEX, namely, SR, PRUM, HiXEval and XCG, can be formulated and calculated using ESR. Two outstanding methodological issues that should be addressed in future work are further refinement on how to assess the relevance of sub-documents [36] and how task-specific issues, such as tolerance to overlap [44], are represented in ESR.

ESR is the first framework of its kind in the literature. ESR measures are comparable with respect to each other because they share a common basis for defining the way they consider relevance, user navigation and redundancy. It provides insights into how measures relate *and* differ, which is not easily replicated with current SDR measures.

We believe that relevance, user navigation and redundancy are also of concern to search tasks outside of SDR. For instance, within the context of semantic web search systems (i.e. searching collections of RDF documents [18]), we are investigating how ESR can be applied to evaluate systems that search collections that do not contain structured documents but, instead, structured information (e.g., semantic associations and ontologies), where navigation also plays an important role. Comparing the relative effectiveness of semantic web search systems using classical precision and recall is a well-known challenge [3, 4, 16]. Our belief is that our ESR framework can serve as a basis to define measures for evaluating search tasks across SDR, semantic web search of RDF collections, and many other areas of information access. Finally, the ESReval package, written in Java, implements all of the measures presented in this work and is available upon request from the authors.

Acknowledgments Financial support for M. S. Ali and Mariano Consens was provided by the Natural Sciences and Engineering Research Council of Canada (NSERC)

References

1. Agichtein, E., Brill, E., Dumais, S., & Ragno, R. (2006). Learning user interaction models for predicting web search result preferences. In *SIGIR '06: Proceedings of the 29th ACM International Conference on Research and Development in IR* (pp. 3–10). New York: ACM.
2. Agrawal, R., Gollapudi, S., Halverson, A., & Ieong, S. (2009). Diversifying search results. In *WSDM '09: Proceedings of the Second International Conference on Web Search and Web Data Mining* (pp.5–14). New York: ACM.
3. Aleman-Meza, B. et al. (2004). Sweto: Large-scale semantic web test bed. In *Proceedings 16th International Conference Software Engineering & Knowledge Engineering, Workshop on Ontology in Action* (pp. 490–493). Knowledge Systems Institute

4. Aleman-Meza, B., Halaschek-Wiener, C., Arpinar, I. B., Ramakrishnan, C., & Sheth, A. P. (2005). Ranking complex relationships on the semantic web. *IEEE Internet Computing, 9*(3), 37–44.
5. Ali, M. S., Consens, M. P., Kazai, G., & Lalmas, M. (2008). Structural relevance: a common basis for the evaluation of structured document retrieval. In *CIKM '08: Proceedings of the 17th ACM Conference on Information and Knowledge Management* (pp. 1153–1162). New York: ACM.
6. Ali, M. S., Consens, M. P., & Lalmas, M. (2007). Structural Relevance in XML Retrieval Evaluation. In *SIGIR 2007 Workshop on Focused Retrieval* (pp. 1–8).
7. Ali, M. S., Consens, M. P., & Larsen, B. (2009). Representing user navigation in xml retrieval with structural summaries. In *ECIR '09*, (pp. 719–723).
8. Amer-Yahia, S., Botev, C., Dörre, J., & Shanmugasundaram, J. (2006). Xquery full-text extensions explained. *IBM Systems Journal, 45*(2), 335–351.
9. Bernstein, Y., & Zobel, J. (2005). Redundant documents and search effectiveness. In *CIKM '05: Proceedings of the 14th ACM Conference on Information and Knowledge Management* (pp. 736–743). New York: ACM.
10. Bollmann, P. (1984). Two axioms for evaluation measures in information retrieval. In *SIGIR '84: Proceedings of the 7th ACM International Conference on Research and Development in IR* (pp. 233–245). British Computer Society.
11. Buckley, C., & Voorhees, E. M. (2000). Evaluating evaluation measure stability. In: *SIGIR '00: Proceedings of the 23rd ACM International Conference on Research and Development in IR* (pp. 33–40). New York: ACM.
12. Consens, M. P., Rizzolo, F., & Vaisman, A. A. (2008). AxPRE Summaries: Exploring the (Semi-)Structure of XML Web Collections. In *ICDE '08: Proceedings of the 24th International Conference on Data Engineering* (pp. 1519–1521). New York: IEEE.
13. Cooper, W. S. (1968). Expected search length: A single measure of retrieval effectiveness based on weak ordering action of retrieval systems. *JASIST, 19*, 30–41.
14. Drosou, M., & Pitoura, E. (2010). Search result diversification. *SIGMOD Record, 39*, 41–47.
15. Dunlop, M. D. (1997). Time, relevance and interaction modelling for information retrieval. In *SIGIR '97: Proceedings of the 20th ACM International Conference on Research and Development in IR* (pp. 206–213). New York: ACM.
16. Fernandez, Lopez, Uren, M., Vallet, Motta, & Castells (2009). Using trec for cross-comparison between classic ir and ontology-based search models at a web scale. In *Semantic Search Workshop, WWW09.* New York, NY: ACM.
17. Fox, S., Karnawat, K., Mydland, M., Dumais, S., & White, T. (2005). Evaluating implicit measures to improve web search. *ACM Transaction on Information System, 23*, 147–168.
18. Guha, R. V., McCool, R., & Miller, E. (2003). Semantic search. In *WWW '03* (pp. 700–709).
19. Hammer-Aebi, B., Christensen, K. W., Lund, H., & Larsen, B. (2006). Users, structured documents and overlap: Interactive searching of elements and the influence of context on search behaviour. In *IIiX '06, Proceedings of the 1st International Conference on Information Interaction in Context* (pp. 46–55). New York: ACM.
20. Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. *ACM Transaction on Information System, 20*(4), 422–446.
21. Kamps, J., Lalmas, M., & Pehcevski, J. (2007). Evaluating relevant in context: document retrieval with a twist. In *SIGIR '07: Proceedings of the 30th ACM International Conference on Research and Development in IR* (pp. 749–750). New York: ACM.
22. Kamps, J., Pehcevski, J., Kazai, G., Lalmas, M., & Robertson, S. (2008). Inex 2007 evaluation measures. In *INEX '07: Focused Access to XML Documents, 6th International Workshop of the Initiative for the Evaluation of XML Retrieval, Selected Papers, LNCS* (vol. 4862, pp. 24–33). Berlin: Springer.
23. Kazai, G. (2007). Choosing an ideal recall-base for the evaluation of the focused task: Sensitivity analysis of the XCG evaluation measures. In *INEX '06: Advances in XML Information Retrieval and Evaluation, 5th International Workshop of the Initiative for the Evaluation of XML retrieval, revised selected papers, LNCS* (vol. 4518, pp. 35–44). Berlin: Springer.
24. Kazai, G., & Lalmas, M. (2006). Extended cumulated gain measures for the evaluation of content-oriented XML retrieval. *ACM Transaction on Information System, 24*(4), 503–542.
25. Kazai, G., Lalmas, M., & Rölleke, T. (2002). Focussed structured document retrieval. In *SPIRE '02: Proceedings of the 9th International Symposium on String Processing and Information Retrieval* (pp. 241–247). Berlin: Springer.
26. Kazai, G., Lalmas, M., & de Vries, A. (2006). Reliability Tests for the XCG and inex-2002 Metrics. In *INEX '05: Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, Revised Selected Papers, LNCS* (vol. 3977, pp. 60–72). Berlin: Springer.

27. Kazai, G., Lalmas, M., & de Vries, A. P. (2004). The overlap problem in content-oriented XML retrieval evaluation. In *SIGIR '04: Proceedings of the 27th ACM International Conference on Research and Development in IR* (pp. 72–79). New York: ACM.
28. Kazai, G., Piwowarski, B., & Robertson, S. E. (2007). Effort-precision and gain-recall based on a probabilistic navigation model. In *ICTIR '07: Proceedings of the 1st International Conference on Theory of Information Retrieval—Studies in Theory of Information Retrieval* (pp. 23–36). Budapest: Foundation for Information Society.
29. Keskustalo, H., Järvelin, K., & Pirkola, A. (2008). Evaluating the effectiveness of relevance feedback based on a user simulation model: effects of a user scenario on cumulated gain value. *Information Retrieval, 11*, 209–228.
30. Ma, H., & Schewe, K. D. (2010). Fragmentation of xml documents. *JIDM, 1*(1), 21–34.
31. Pehcevski, J., & Thom, J. A. (2006). HiXEval: Highlighting XML retrieval evaluation. In *INEX '05: Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, Revised Selected Papers, LNCS* (vol. 3977, pp. 60–72). Berlin: Springer.
32. Pehcevski, J., & Thom, J. A. (2007). Evaluating Focused Retrieval Tasks. In *SIGIR 2007 Workshop on Focused Retrieval* (pp. 33–40).
33. Piwowarski, B. (2006). Eprum metrics and inex 2005. In *INEX '05: Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, Revised Selected Papers, LNCS* (vol. 3977, pp. 30–42). Berlin: Springer.
34. Piwowarski, B., & Gallinari, P. (2005) Expected ratio of relevant units: A measure for structured information retrieval. In *INEX '04: Advances in XML Information Retrieval and Evaluation, 3rd International Workshop of the Initiative for the Evaluation of XML Retrieval, Revised Selected Papers, LNCS* (vol. 3493, pp. 15–17). Berlin: Springer.
35. Piwowarski, B., Gallinari, P., & Dupret, G. (2007). Precision Recall with User Modeling (PRUM): Application to structured information retrieval. *ACM Transaction on Information System, 25*(1), 1.
36. Piwowarski, B., Trotman, A., & Lalmas, M. (2008). Sound and complete relevance assessment for xml retrieval. *ACM Transaction on Information System, 27*(1), 1–37.
37. Raghavan, V., Bollmann, P., & Jung, G. (1989). A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transaction on Information System, 7*(3), 205–229.
38. Robertson, S. (1969). The parametric description of retrieval tests. part 1: The basic parameters; part 2: overall measures. *Journal of Documentation, 25*(1), 1–27, 93–107.
39. Salton, G., Allan, J., & Buckley, C. (1993). Approaches to passage retrieval in full text information systems. In *SIGIR '93: Proceedings of the 16th ACM International Conference on Research and Development in IR* (pp. 49–58). New York: ACM.
40. Sigurbjörnsson, B., Kamps, J., & de Rijke, M. (2005). Mixture Models, Overlap, and Structural Hints in XML Element Retrieval. In *INEX '04: Advances in XML Information Retrieval and Evaluation, 3rd International Workshop of the Initiative for the Evaluation of XML Retrieval, Revised Selected Papers, LNCS* (vol. 3493, pp. 196–210). Berlin: Springer.
41. Trotman, A., & Geva, S. (2006). Report on the SIGIR 2006 workshop on XML element retrieval methodology. *SIGIR Forum, 40*(2), 42–48.
42. Trotman, A., & Lalmas, M. (2006). Strict and vague interpretation of XML-retrieval queries. In *SIGIR '06: Proceedings of the 29th ACM International Conference on Research and Development in IR* (pp. 709–710). New York: ACM.
43. Trotman, A., & Sigurbjörnsson, B. (2006). Narrowed Extended XPath I (NEXI). In *INEX '05: Advances in XML Information Retrieval and Evaluation, 4th International Workshop of the Initiative for the Evaluation of XML Retrieval, Revised Selected Papers, LNCS* (vol. 3977, pp. 16–40). Berlin: Springer.
44. de Vries, A.P., Kazai, G., & Lalmas, M. (2004). Tolerance to irrelevance: A user-effort oriented evaluation of retrieval systems without predefined retrieval unit. In *RIAO '04: Proceedings of the International Conference on Recherche d'Information Assiste par Ordinateur* (pp. 463–473).