

Learning to Summarise XML Documents using Content and Structure

Massih R. Amini[†] Anastasios Tombros* Nicolas Usunier[†] Mounia Lalmas* Patrick Gallinari[†]
amini@poleia.lip6.fr tassos@dcs.qmul.ac.uk usunier@poleia.lip6.fr mounia@dcs.qmul.ac.uk gallinari@poleia.lip6.fr

[†]University Pierre and Marie Curie
8, rue du capitaine Scott
75015, Paris
France

*Queen Mary, University of London
Mile End Road
London E1 4NS
United Kingdom

ABSTRACT

Documents formatted in eXtensible Markup Language (XML) are becoming increasingly available in collections of various document types. In this paper, we present an approach for the summarisation of XML documents. The novelty of this approach lies in that it is based on features not only from the content of documents, but also from their logical structure. We follow a sentence extraction-based summarisation method that employs a novel machine learning approach. To find which features are more effective for producing summaries this approach views sentence extraction as an ordering task. We evaluated our summarisation model using the INEX and SUMMAC datasets. The results demonstrate that the inclusion of features from the logical structure of documents increases the effectiveness of the summariser, and that the novel machine learning approach is also effective and well-suited to the task of summarisation in the context of XML documents. Our approach is generic and is therefore applicable to elements of varying granularity within the XML tree. We view these results as a step towards the intelligent summarisation of XML documents.

Categories and Subject Descriptors

H.4.m [Information Systems]: [Miscellaneous-Text Summarisation]; I.2.6 [Computing Methodologies]: [Learning-Parameter]

General Terms

Algorithms, Experimentation, Performance

Keywords

Summarisation, machine learning, ranking algorithms, XML documents, content and structure features

1. INTRODUCTION

With the growing availability of on-line text resources, it has become necessary to provide users with systems that obtain answers

to queries in a manner which is both efficient and effective. Single document text summarisation (SDS) can be coupled with conventional search engines and help users to evaluate the relevance of documents [32] for providing answers to their queries.

Since its beginnings back in the 1950s [22], automatic text summarisation has been performed primarily by the selection of sentences (or text spans of other sizes, e.g. paragraphs) from the original document; scores are then assigned to sentences according to a set of extraction criteria, and the best-scoring sentences are presented in the summary. Criteria often used in work in extractive summarisation include cue-phrases and positional indicators [22, 11], lexical occurrence statistics [5], title-keyword similarity [11], and the use of discourse structure [24]. This approach can be better described as sentence extraction, rather than summarisation, and although it does not perform an in-depth analysis of the source text it can produce summaries that have proven effective in various information retrieval (IR) tasks [32, 19, 26]. In order to produce human quality summaries, summarisation systems need to borrow elements from fields such as linguistics, discourse understanding and language generation [25]. This approach to summarisation is highly complex, and extraction-based approaches are more common.

Over the last few years there has also been a considerable increase in the availability of documents in XML format [27]. In XML documents a tree-like structure, which corresponds to the logical structure of the source documents, is encoded. For example, an article can be seen as corresponding to the root of the tree, and sections, subsections and paragraphs can be arranged in branches and leaves of the tree.

The content-based retrieval of XML documents has also received interest over the last few years, mainly through the INEX initiative [15]. In XML retrieval, document components, rather than entire documents, are retrieved. As the number of XML components is typically large (much larger than that of documents), it is essential to provide users of XML IR systems with overviews of the contents of the retrieved elements. The summarisation of XML documents is beginning to draw attention from researchers [1, 21, 27]. A major aim of this paper is to investigate the effectiveness of an XML summarisation approach by combining structural and content features to extract sentences for summaries.

In this paper we follow a summarisation approach that is based on the machine learning (ML) paradigm. Like most previous work on ML for summarisation [17, 6, 8], we rely on supervised learning, where a set of training documents and their extract summaries is required. In this work, we use the datasets of the INitiative for the Evaluation of XML retrieval (INEX) [15] and the Computation and

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 200X ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

Language collection (cmp-1g) of TIPSTER SUMMAC [28].

Most previous work on ML for summarisation has followed the classification paradigm [17, 30, 8, 2]. In such approaches, a classifier is trained to distinguish between two classes of sentences: summary and non-summary ones. The classifier is learned by comparing its output to a desired output reflecting global class information. This framework is limited in that it makes the assumption that all sentences from different documents are comparable with respect to this class information.

In this work, we explore new ML approaches for SDS based on the Area under the ROC curve (AUC). The main rationale of this approach is to automatically combine different features, each being a numerical representation of a given extraction criterion. The summariser learns how to best combine sentence features based on its effectiveness at assigning higher scores to summary sentences than to non-summary sentences. This ordering criterion corresponds exactly to what the learnt function is used for, i.e. ordering sentences.

The contributions of this work are therefore twofold: first, we propose and justify the effectiveness of a new algorithm that optimises the AUC ordering criterion, instead of the mostly used classification error criterion in ML approaches for SDS, and second, we investigate the summarisation of XML documents by taking into account features relating both to the content and the logical structure of the documents.

The ultimate aim of our approach is to generate summaries for components of XML documents at any level. Since at present the evaluation of such summaries is hard (due to the lack of appropriate resources), we consider an XML article to be an XML element, and we use its content and structure to learn how we can best summarise it. Our approach is sufficiently generic to be applied to a component at any level of the logical structure of an XML document.

In the remaining of the paper, we first discuss related work on ML approaches for optimising ordering criteria, and previous work on the summarisation of XML documents in section 2. Then, in section 3, we outline our ML approach for summarising XML documents. In section 5 we present the results of our evaluation, and in section 6 we discuss the outcomes of this study and we also draw some pointers for taking this work further.

2. BACKGROUND

In this section we present the main motivation for using new ML approaches based on ordering criteria for summarisation, and we discuss some previous work on the summarisation of XML documents.

2.1 Text summarization as a classification task

In order to combine sentence features, ML approaches for SDS adopt a classification framework, like the Naive Bayes model [17] or a logistic regression classifier [2]. The working principle of classification approaches to SDS is to associate class label 1 to summary (or *relevant*) sentences, and class label -1 to non-summary (or *irrelevant*) ones, and to use the learning algorithm to discover the weights of a linear combination of the features. Such approaches estimate the probability of a sentence being in the appropriate class, with the goal of optimizing a classification criterion. The scoring function used to score a sentence of a new document is the weighted sum of its features, where the weights are learnt by the classification algorithm. Sentences are then ranked with respect to the output of the classifier and those with the highest scores are extracted to form the summary of the document.

For example, in the particular case of a logistic classifier, one makes the following assumption on the form of the posterior probability of the class *relevant* given a sentence s represented by a

vector of scores (s_1, \dots, s_n) :

$$p(\text{relevant} | s) = \frac{1}{1 + e^{-2 \sum_{i=1}^n \lambda_i s_i}}$$

The parameters of the feature combination $\Lambda = (\lambda_1, \dots, \lambda_n)$ can then be learnt by maximizing the binomial log-likelihood:

$$\mathcal{L}(\mathcal{D}; \Lambda) = -\frac{1}{2} \sum_{y=-1,1} \frac{1}{|\mathcal{S}^y|} \sum_{s \in \mathcal{S}^y} \log(1 + e^{-2y \sum_{i=1}^n \lambda_i s_i}) \quad (1)$$

where \mathcal{D} is the set of training documents, \mathcal{S}^1 and \mathcal{S}^{-1} are respectively the set of relevant and irrelevant sentences in the training set and $|\mathcal{S}^y|$ is the cardinal of the set \mathcal{S}^y for $y \in \{-1, 1\}$.

2.2 Text summarisation as an ordering task

We propose here a novel framework for training a ML based text summarizer by optimizing an *ordering* criterion. We build upon the work of [3] who proposed a ranking framework for SDS and proved empirically that a ranking algorithm outperforms a classifier. Here, we go further by giving a general ordering framework for SDS based on the Area under the ROC curve (AUC) [4, 36]. This framework allows a formal justification of why an ordering criterion is better suited for SDS than a classification one.

2.2.1 Formal justification of ordering for SDS

In [9] it has been theoretically shown that the standard deviation of the AUC measure at a given error rate is high when the error rate is high itself, showing that optimising the error rate of a classifier would not yield an optimisation of its AUC. Furthermore, Caruana et al. have empirically shown, on a large number of experiments, that standard optimisation criteria for classification are lowly correlated to the classifier performance in terms of the AUC [7].

In conclusion, a good summariser should have high ordering performance, which can, for example, be measured via the AUC, while the optimisation of the standard classification criterion yields a possibly low ordering performance. It follows that classification algorithms trained by optimising a standard classification criterion may not be well suited to the task of text summarisation.

For SDS, let us consider pairs of sentences (s, s') , where one of the sentences is relevant and the other one irrelevant. An algorithm optimising the AUC, would train a scoring function H by minimising a classification error over the considered pairs of sentences: a pair (s, s') is given the correct class label if H assigns a higher score to the relevant sentence in the pair than to the irrelevant one. Considering such pairs of sentences allows such an algorithm to focus on the relative scores of relevant/irrelevant sentences, while standard classification algorithms compare the absolute scores of the sentences with a given threshold.

2.2.2 Optimising the AUC metric

There has been a surge of interest for the AUC in the last few years in the machine learning community. In particular, several different optimisation methods for directly optimising the AUC have emerged, via approximations [16, 38] or upper bounds over the Wilcoxon-Mann-Whitney statistic [9].

Using the same notation as in section 2.1, the AUC for a scoring function H is equal to (see e.g. [9]):

$$\begin{aligned} A &= \frac{1}{|\mathcal{S}^{-1}| |\mathcal{S}^1|} \sum_{(s, s') \in \mathcal{S}} ([[H(s') < H(s)]] + \frac{1}{2} [[H(s) = H(s')]]) \\ &\geq 1 - \frac{1}{|\mathcal{S}^{-1}| |\mathcal{S}^1|} \sum_{(s, s') \in \mathcal{S}^1 \times \mathcal{S}^{-1}} [[H(s') \geq H(s)]] \end{aligned}$$

where $S = S^1 \times S^{-1}$ and $[[\pi]]$ is equal to 1 if predicate π is true and 0 otherwise.

Maximising the AUC can then be performed by minimising A' :

$$A' = \frac{1}{|S^{-1}||S^1|} \sum_{(s,s') \in S^1 \times S^{-1}} [[H(s') \geq H(s)]] \quad (2)$$

For the case of SDS, the quantity under the summation computes, over all documents of the training set, the number of irrelevant sentences that have been scored higher than the relevant ones by H . The learning resumes by finding the optimal combination of sentence features which minimises equation (2).

2.3 Summarising XML documents

There have been few researchers that have investigated the summarisation of information available in XML format. In [1], the work focuses on retaining the structure of the source document in the summary. A textual summary of a document is created by using lexical chains. The textual summary is then combined with the overall structure of the document with the aim of preserving the structure of the original document and of superimposing the summary on that structure. In [27], the idea of generating semantic thumbnails (essentially summaries) of documents in XML format is suggested. The authors propose to utilise the ontologies embedded in XML and RDF documents in order to develop the semantic thumbnails. Litkowski [21] has used some discourse analysis of XML documents for summarisation. In some other work [10], the tree representation of XML documents is used to generate tree structural summaries; these are summaries that focus on the structural properties of trees and do not correspond to summaries in the conventional sense of the term as used in IR research. Operations such as nesting and repetition reduction in the XML trees are used.

In the above approaches, features pertaining to the logical structure of XML documents are not taken into account when producing summaries. Structural clues are used by work on summarisation of other document types, e.g. e-mails [18], or technical documents [35]. In these summarisation approaches, known features of the structure of documents are exploited in order to produce summaries (e.g. the presence of a FAQ, or a question/answer section in technical documents).

In this paper, we take the logical structure of documents into account when producing summaries, as well as the content, and we learn an effective combination of features for summarisation. Although for evaluation purposes we use the INEX and SUMMAC collections, which contain scientific articles, our approach could apply to any documents formatted in XML where the logical structure is available. The summarisation of scientific texts through sentence extraction has been extensively studied in the past [31]. In our approach, we do not explicitly take advantage of the idiosyncratic nature of scientific articles, but we rather propose a generic approach that is, in essence, genre-independent. In the next section we present the specific details of our methodology.

3. APPROACH

In this section we first discuss the learning algorithm for optimising the *ordering* AUC criterion, then we outline the features of XML documents that we employed in our summarisation model, and then we present the details of the evaluation.

3.1 The learning algorithm

Based on the same logistic model as the one presented in section 2.1 for classification, we propose here a learning algorithm for SDS, optimising the AUC metric. This will constitute a fair

working scheme for comparing these two frameworks as the only difference here resides in the learning criteria.

We represent the pair (s, s') of sentences by the difference of their representative vectors, $(s_1 - s'_1, \dots, s_n - s'_n)$. The logistic assumption in this case writes:

$$p(\text{relevant} | s, s') = \frac{1}{1 + e^{-2 \sum_{i=1}^n \theta_i (s_i - s'_i)}} \quad (3)$$

The parameters $\Theta = (\theta_1, \dots, \theta_n)$ are used to score any sentence s in the collection by $H(s) = \sum_{i=1}^n \theta_i s_i$ which is a linear combination of the input features (section 3.2). As the ordering criterion (2) is not differentiable, following the method for ranking proposed by [13], the parameters of the combination Θ can be obtained by minimising an exponential upper bound on A' :

$$\mathbf{EAUC}(\mathcal{D}; \Theta) = \frac{1}{|S^{-1}||S^1|} \sum_{(s,s') \in S^1 \times S^{-1}} e^{H(s') - H(s)} \quad (4)$$

The binomial log-likelihood for AUC writes:

$$\mathcal{L}_{\mathcal{A}}(\mathcal{D}; \Theta) = -\frac{1}{|S^{-1}||S^1|} \sum_{(s,s') \in S} \log(1 + e^{-2(H(s) - H(s'))}) \quad (5)$$

Following [14], one can asymptotically show that the population minimizers of the expected binomial log-likelihood for AUC and $Ee^{H(s') - H(s)}$ coincide. This is an interesting finding which reinforces the duality between classification and the present framework. An interesting property of the exponential loss, \mathbf{EAUC} , is that it can be computed in time linear to the number of examples, simply by rewriting the equation (4) as:

$$\mathbf{EAUC}(\mathcal{D}; \Theta) = \frac{1}{|S^{-1}||S^1|} \left(\sum_{s' \in S^{-1}} e^{H(s')} \right) \left(\sum_{s \in S^1} e^{-H(s)} \right) \quad (6)$$

The optimisation criterion (4) is a convex function, so standard optimisation algorithms can be used to minimise it. In our case, we used an iterative scaling algorithm to learn the parameters of the logistic model (3), which is an adaptation for AUC of an algorithm developed in [20].

Compared to the state-of-the-art, our approach bears similarity with the *RankBoost* algorithm [13] proposed for ranking. In both cases the parameters of the combination are learned by minimising a convex function. However, the main difference is that we propose here to learn a linear combination of the features by directly optimising (4) through a standard optimisation algorithm, while *RankBoost* learns iteratively a nonlinear combination of the features by adaptively resampling the training data. Another advantage of our approach is that it allows to compute a tight generalisation error bound from the training set, and hence enables us to draw practical conclusions on learning algorithms which optimize the AUC [34].

3.2 Document features for summarisation

The ranking algorithm learns features both from the content and the logical structure of the XML documents.

3.2.1 Content features

Terms contained in the title of a document have long been recognised as effective features for automatic summarisation [11]. Our basic content-only query (COQ) comprises terms in the title of the document (*Title* query), as well as the title keywords augmented by the most frequent terms in the document (up to 10 such terms) (*Title-MFT* query). The rationale of these approaches is that these terms should appear in sentences that are worthwhile including in summaries. The importance of title terms for SDS can also be

extended to components of finer granularity (e.g. sections, sub-sections, etc.), by using the title of the document to find relevant sentences within any component, or, where appropriate, by using meaningful titles of components.

Since the *Title* query may be very short, we have also employed query-expansion techniques such as Local Context Analysis (LCA) [37] or thesaurus expansion methods (i.e. WordNet [12]), as well as a learning based expansion technique.

3.2.1.1 Expansion via WordNet and LCA.

From the *Title* query, we formed two other queries, reflecting local links between the title keywords and other words in the corresponding document:

- *Title-LCA* query, includes keywords in the title of a document and the words that occur most frequently in sentences that are most similar to the *Title* query according to the cosine measure.
- *Title-WN*, includes expanded title keywords and all their first order synonyms using WordNet.

We used the cosine measure in order to compute a preliminary score between any sentence of a document and these four queries (Title, Title-MFT, Title-LCA, Title-WN). The scoring measure doubles the cosine scoring of sentences containing acronyms (e.g. HMM (Hidden Markov Models), NLP (Natural Language Processing)), or cue-terms, e.g. "in this paper", "in conclusion", etc. The use of acronyms and cue phrases in summarisation has been emphasised in the past by [11, 17].

3.2.1.2 Expansion with word-clusters.

We also included two queries using word clusters. This is another source of information about the relevance of sentences to summaries. It is a more contextual approach compared to the title-based queries, as it seeks to take advantage of the co-occurrence of terms within sentences all over the corpus, as opposed to the local information provided by the title-based queries.

We form different word-clusters based on words co-occurring in sentences of all documents in the corpus. For discovering these word-clusters, each word w in the vocabulary V is first characterised as a p -dimensional vector $\mathbf{w} = \langle n(w, i) \rangle_{i \in \{1, \dots, p\}}$ representing the number of occurrences of w in each sentence i . Under this representation, word clustering is performed using the Naive-Bayes clustering algorithm maximising the Classification Maximum Likelihood criterion [29]. We have arbitrarily fixed the number of clusters to $\frac{|V|}{100}$.

From these clusters we obtained two expanded queries. First, we added to title keywords words that occurred in their respective clusters, *Extended concepts with word clusters* query. Second, we projected each sentence of a document and the *Title* query in the space of these word-clusters, *Projected concepts on word clusters* query. For the latter query, we represent the *title* query, and each sentence in a document, by a vector where each dimension of the vector represents the number of occurrences of words from a cluster in that sentence or in the *title* query. The dimensions in this representation are related to the degree of representation of each word-cluster in a given sentence, or in the *title* query.

Table 1 shows some term clusters found for the SUMMAC data collection; it can be seen from this example that each cluster can be associated to a general concept.

3.2.2 Structural features

Past work on SDS (e.g. [11, 17]) has implicitly tried to take the structure of certain document types into account when extracting

Term Clusters	
Cluster i:	transduction language grammar set word information model number words rules rule lexical
Cluster j:	tag processing speech recognition morphological korean morpheme

Table 1: An example of term clusters found for the SUMMAC data collection.

sentences. In [17], for example, the leading and trailing paragraphs in a document are considered important, and the position of sentences within these paragraphs is also recorded and used as a feature for summarisation.

In our work, we move into an explicit use of structural features by taking into account the logical structure of XML documents. Our aim is to investigate whether taking the structure of documents (or document components) into account when generating the summaries is more beneficial than using the content features alone.

The structural features we use in our approach are:

1. The depth of the element in which the sentence is contained (e.g. section, subsection, subsubsection, etc.).
2. The sibling number of the element in which the sentence is contained (e.g. 1st, middle, last).
3. The number of sibling elements of the element in which the sentence is contained.
4. The position in the element of the paragraph in which the sentence is contained (e.g. first, or not).

These features are generic, and can be applied to an entire document, or to components at any level of the XML tree; they are also applicable to document collections of different genres. These are just some of the features that can be used for modeling structural information; many of them have been considered for example in XML retrieval approaches (see [15]). In future work, we intend to experiment with a bigger set of structural features.

4. EXPERIMENTS

In our experiments, we used 2 data sets - INEX [15] and SUMMAC [28] test collections. For each dataset, we carried out two evaluation experiments testing the query expansion effects (sections 3.2.1.1 and 3.2.1.2) and the learning effect. For the latter, we ran 3 algorithms - a logistic model optimising the ordering AUC metric (4) using an iterative scaling scheme [20], a logistic classifier optimising the classification binomial log-likelihood criteria (1) and the RankBoost algorithm [13]. To measure the effect of structure features we have learned the best learning algorithm using COQ features alone and COQ features with the aforementioned structure features.

4.1 Datasets

The INEX document collection consists of 12,107 articles of the IEEE Computer Society's publications, from 1995 to 2002, totaling 494 megabytes. The collection contains over 8.2 million element nodes of varying granularity, where the average depth of a node is 6.9 (taking an article as the root of the tree). The overall structure of a typical article consists of a frontmatter (containing e.g. title, author, publication information and abstract), a body (consisting of e.g. sections, sub-sections, sub-sub-sections, paragraphs, tables, figures, lists, citations) and a backmatter (including bibliography and author information).

The SUMMAC corpus consists of 183 articles. Documents in

this collection are scientific papers which appeared in ACL sponsored conferences. The collection has been marked up in XML by converting automatically the latex version of the papers to XML. In this dataset the markup includes tags covering information such as title, authors or inventors, etc., as well as basic structure such as abstract, body, sections, lists, etc.

We have also removed documents, from the INEX dataset, deprived of title keywords or an abstract. From the SUMMAC dataset, we removed documents whose title contained no-informative words such as a list of proper names. From each dataset, we also removed documents having extractive summaries (found by Marcu’s algorithm, see section 4.2) composed of one sentence only, arguing that a sentence is not sufficient to summarise a scientific article. In our experiments, we used in total 161 documents from SUMMAC and 4446 documents from INEX collections.

We extracted the logical structure of XML documents using freely available structure parsers. Documents are tokenised by removing words in a stop list, and sentence boundaries within each document are found using the morpho-syntactic tree tagger program [33].

In Table 2 we display some statistics about the two document collections used, about the abstracts provided with the two collections, and about the extracts that were created using Marcu’s algorithm.

Data set comparison		
source	SUMMAC	INEX
Number of docs	161(183)	4446(12107)
Average # of sent. per doc.	153.57	183.8
Maximum # of sent. per doc.	799	864
Minimum # of sent. per doc.	6	12
Average # of words per sent.	11.39	14.12
Size of the vocabulary	15119	153422
Average extract size (in # of sent)	9.71	6.07
Maximum # of sent. per extract	36	37
Minimum # of sent. per extract	2	3
Average abstract size (in # of sent)	9.2	5.92
Maximum # of sent. per abstract	24	25
Minimum # of sent. per abstract	4	5

Table 2: Data Set properties

4.2 Evaluation

We assume that for each document, summaries will only include sentences between the introduction and the conclusion of the document. A compression ratio must be specified for extractive summaries. For both datasets we followed the SUMMAC evaluation by using a 10% compression ratio [28].

To obtain sentence-based extract summaries for all articles in both datasets, for training and evaluation purposes, we need *gold* summaries. The human extraction of such reference summaries, in the case of large datasets, is infeasible. To overcome this restriction we use in our experiments the author-supplied abstracts that are available with the original articles, and apply an algorithm proposed by Marcu [23] in order to generate extracts from the abstracts. This algorithm has shown a high degree of correlation to sentence extracts produced by humans. We therefore evaluate the effectiveness of our learning algorithm on the basis of how well it matches the automatic extracts.

The learning algorithms take as input the set of features that we defined in the previous sections. Each sentence in the training set is represented as a feature vector, and the algorithms are learned based on this input representation and the extracted summaries found by Marcu’s algorithm [23], which were used as desired outputs.

For all the algorithms, on each dataset, we have generated Precision and Recall curves with 10 fold cross-validation (Table 3). Precision and Recall are computed as follows:

$$\text{Prec} = \frac{\# \text{ of sentences in the extract and also the gold standard}}{\text{total \# of sentences in the extract}}$$

$$\text{Recall} = \frac{\# \text{ of sentences in the extract and also the gold standard}}{\text{total \# of sentences in the gold standard}}$$

We have also measured the break-even point at 10% compression ratio for the 3 learning algorithms and the best COQ feature (Table 4).

5. ANALYSIS OF RESULTS

We examine the results from three viewpoints: in section 5.1 we present the effectiveness of each content only queries (COQ) alone as well as the query expansion effect, in section 5.2 we examine the performance of the three learning algorithms, and in section 5.3 we look into the effectiveness of our summarisation approach for XML documents.

5.1 Query expansion effects

In figures (a and b) in Table 3, we present data for the effectiveness of content only features for SDS without the learning effect (i.e. by using each content feature individually to rank the sentences). The order of effectiveness of the features seems to be consistent across the two datasets: extended concepts with word clusters are the most effective, followed by projected concepts on word clusters and title with local context analysis. Title with the most frequent terms in the document is the least effective feature in both cases.

The high effectiveness obtained with word clusters (extended and projected concepts with word clusters) demonstrates that the more contextual approach of forming word clusters by utilising the co-occurrence of terms within sentences of an entire corpus is effective and should be further exploited for SDS.

5.2 Learning algorithms

In Table 3 (figures a’ and b’) we present the precision and recall graphs that we obtained through the combination of content and structure features for the two datasets when using the three learning algorithms.

A first, non-surprising, result is that the combination of features by learning outperforms each feature alone. The results also show that the two ordering algorithms are more effective on both datasets than the logistic classifier. However, the comparison between the AUC algorithm and the logistic classifier leads to the conclusion that an ordering criterion is better suited to SDS than a classification criterion. This finding corroborates with the justification given in section 2.2.1.

When comparing the two ordering algorithms, we see that the AUC algorithm slightly outperforms the RankBoost algorithm for high recall values. Since both ordering algorithms optimise the same criteria, equation (4), the difference in performance can be explained by the class of functions that each algorithm learns. The RankBoost algorithm outputs a nonlinear combination of the features while with the AUC algorithm we obtain a linear combination of these features. As the space of features is small, the non-linear RankBoost model has low bias and high variance and hence attempts to overfit the data. We have noticed this effect on both test collections by comparing Precision and Recall curves for RankBoost on the test and the training sets.

Our experimental results suggest that the AUC criterion suits the SDS task better than a classification criterion. Moreover, a simple

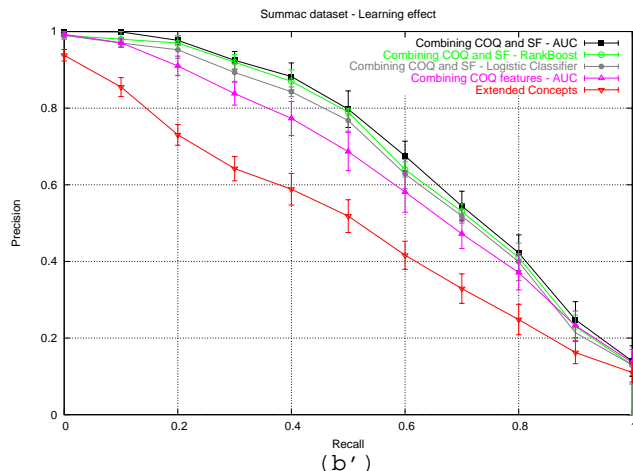
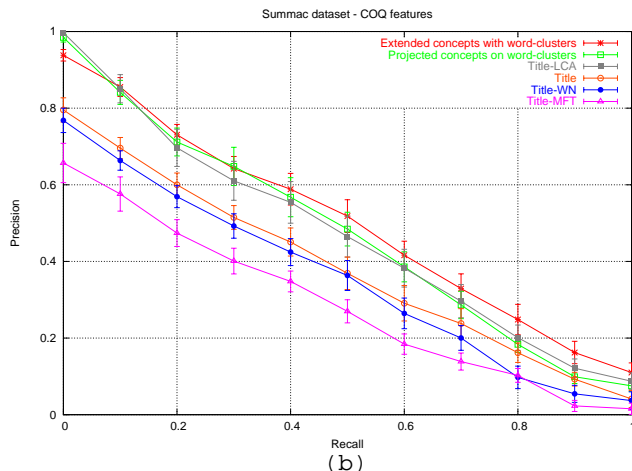
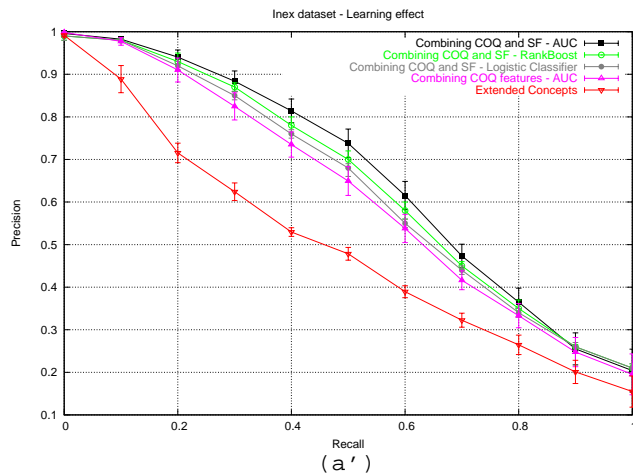
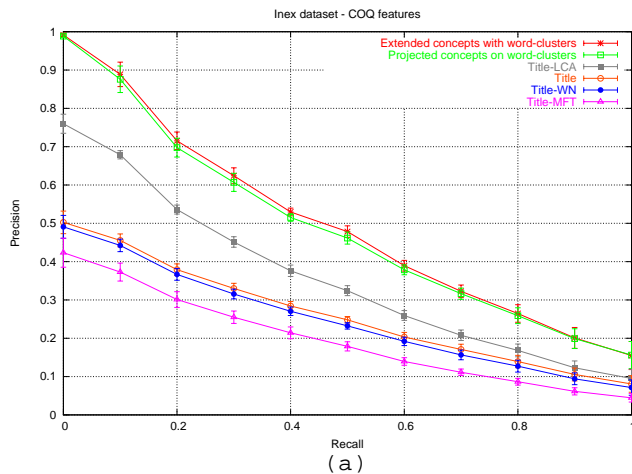


Table 3: Precision-Recall curves at 10% compression ratio for the COQ features ((a) - (b)) and the learning effects ((a') - (b')) on INEX (top) and SUMMAC (down) datasets. Each point represents the mean performance for 10 cross-validation folds. The bars show standard deviations for the estimated performance. The combination of COQ and structure features outperforms the combination of COQ features alone.

logistic model performs better than a non-linear algorithm and, depending on the implementation, can be significantly faster to train than RankBoost. This leads to the conclusion that such a linear model, i.e. optimising the AUC, can be a good choice for learning a summariser, in particular when considering structural features.

5.3 Summarisation effectiveness

By looking at the data in Table 3 (figures a' and b') from the point of view of comparing the effectiveness of the summariser with different features, one can note that the combination of content and structure features yields greater effectiveness than the use of content features alone. This result seems to hold equally for both document sets for most recall points. In terms of break-even points, (Table 4), the increase in effectiveness is approximately +3% for RankBoost and AUC algorithms on both data sets¹. This provides evidence that the use of structural features improves the effectiveness of the task of SDS.

¹The same performance increase is also obtained from the classifier.

From the set of structure features we used in our experiments (section 3.2.2), the depth of the sentence's component and the paragraph's position containing summary sentences within the component (i.e. whether it is in the first paragraph or not of a component) are the two most effective for SDS. It is well known that, in scientific articles, sentences in the first parts of sections such as Introduction and Conclusions are useful for summarisation purposes [11, 17]. Our results agree with this, as the increased weights for the paragraph's position in a component suggests. The features corresponding to the position of elements with respect to their siblings had less effect than the two previous ones, but features indicating the position of an element as the first or the last sibling had a higher impact than when the element was the middle sibling. We should also note that the feature corresponding to the number of siblings of an element was the least conclusive in all of our experiments; its utility seemed to highly depend on the dataset.

For the specific case of scientific text, from the set of structure features used, a set of features which is known to be effective was weighted higher by our summarisation method. This agreement can

Precision-Recall break-even point (%)							
Data sets	Best COQ feature	Classifier		RankBoost		AUC	
		COQ features	COQ+SF	COQ features	COQ+SF	COQ features	COQ+SF
SUMMAC	50.9	56.4	61.05	57.8	62.1	59.1	63.2
INEX	48.5	55.1	58	56.1	59.1	57.06	60.6

Table 4: Break-even points at 10% compression ratio for learning algorithms and the best COQ feature: Extended title keywords with word-clusters. Each value represents the mean performance for 10 cross-validation folds.

be seen as an indication that the use of structure features could be applied to document collections of different genre. The availability of suitable document collections containing different document types will be necessary in order to test this assertion.

By looking at the data in Table 4 (and in the figures of Table 3), one can note that effectiveness when using the INEX collection is always lower than when using the SUMMAC collection. This difference in effectiveness can be attributed to the different characteristics of the two datasets. The INEX collection contains much more documents than SUMMAC, and is also a more heterogeneous dataset. In addition, the logical structure of INEX documents is more complex than those of the SUMMAC collection. These factors are likely to have caused the small difference in effectiveness between the two collections.

6. DISCUSSION AND CONCLUSIONS

The results that we presented in the previous section are encouraging in relation to our two main motivations: a novel learning algorithm for SDS, and the inclusion of structure, in addition to content, features for the summarisation of XML documents. In terms of the algorithms, it was shown that using the same logistic model, but choosing an ordering criterion instead of a classification one, leads to a notable performance increase. Moreover, compared to RankBoost, the algorithm learnt with the AUC criterion performs better and it also has the potential to be implemented in a simpler manner. This property may make the AUC algorithm an effective and efficient choice for the task of SDS.

In terms of the summarisation of XML documents by using content and structure features, the results demonstrated that for both datasets, the inclusion of structural features improved the effectiveness of learning algorithms for SDS. The improvements were not dramatic, but they were consistent across both datasets and across most recall points. This consistency suggests that the inclusion of features from the logical structure of XML documents is effective.

The ultimate aim of our approach for the summarisation of XML documents is to produce summaries for components at any level of granularity (e.g. section, subsection, etc.). The content and structure features that we presented in section 3.2 can be applied to any level of granularity. In particular, the most effective content (expanded concepts with word clusters and projected concepts on word clusters), and structure features (depth of element and position of paragraph in the element), can be applied to various granularity levels within an XML tree. The effectiveness of such an approach however, can not be tested until datasets with human produced summaries, or summary extracts, at component level become available. We should also note that we focus on generic (rather than query-biased) summaries for evaluation purposes, but the proposed model can be applied to both types of summarisation.

In section 5.3 we mentioned that the results provide us with some indication that the use of structural features can also be effective for summarising XML documents from datasets containing documents other than scientific articles. One possible direction for future research would therefore be to examine this issue in more detail.

The list of structural features that we use in this study is short, so a larger variety of features could be investigated. When moving into document collections of different types, it will be worthwhile to investigate whether useful structural features can be derived automatically, e.g. by looking at a collection’s DTD.

Some further interesting issues that arise when considering the summarisation at any structural level relate to the choice of the appropriate components to be summarised. For example, it may be unrealistic to provide summaries of very small size components, or of components that are not informative enough. One of the main research issues in XML retrieval is to define and understand what a meaningful retrieval unit is [15]. One possible direction to follow, would be to conduct a user study in which to observe what kinds of XML elements searchers would prefer to see in a summarised version after the initial retrieval.

By looking at the results of this study as a whole, we can say that the work presented here achieved its main aim, to effectively summarise XML documents by combining content and structure features through using novel machine learning approaches. Both datasets that we used contain scientific articles, that have some inherent characteristics which may simplify the task of SDS. This work has however a greater impact, as we believe that it can be applied to datasets containing documents of other types. The availability of XML data will continue to increase as, for example, XML is becoming the W3C standard for representing documents (e.g. in digital libraries where content can be of any type). The availability of intelligent summarisation approaches for XML data with therefore become increasingly important, and we believe that this work has provided a step towards this direction.

7. REFERENCES

- [1] H. Alam, A. Kumar, M. Nakamura, F. Rahman, Y. Tarnikova, and C. Wilcox. Structured and unstructured document summarization: Design of a commercial summarizer using lexical chains. In *Proceedings of the 7th International Conference on Document Analysis and Recognition*, pages 1147–1152. IEEE, 2003.
- [2] M.-R. Amini and P. Gallinari. The use of unlabeled data to improve supervised learning for text summarization. In *Proceedings of the 25th ACM SIGIR Conference*, pages 105–112, 2002.
- [3] M.-R. Amini, N. Usunier, and P. Gallinari. Automatic text summarization based on word clusters and ranking algorithms. In *Proceedings of the 27th European Conference on Information Retrieval*, pages 142–156, 2005.
- [4] D. Bamber. The area above the ordinal dominance graph and the area below the receiver operating characteristic graph. *Journal of Mathematical Psychology*, 12:387–415, 1975.
- [5] R. Barzilay and M. Elhadad. Using lexical chains for text summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS’97)*, ACL, 1997.
- [6] A. L. Berger and V. O. Mittal. Ocelot: a system for summarizing web pages. In *Proceedings of the 23rd ACM*

- SIGIR Conference*, pages 144–151, 2000.
- [7] R. Caruana and A. Niculescu-Mizil. Data mining in metric space: an empirical analysis of supervised learning performance criteria. In *Proceedings of the 10th ACM SIGKDD Conference*, pages 69–78, 2004.
- [8] W. T. Chuang and J. Yang. Extracting sentence segments for text summarization: a machine learning approach. In *Proceedings of the 23rd ACM SIGIR Conference*, pages 152–159, 2000.
- [9] C. Cortes and M. Mohri. AUC optimization vs. error rate minimization. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.
- [10] T. Dalamagas, T. Cheng, K.-J. Winkel, and T. K. Sellis. Clustering xml documents using structural summaries. In *Proceedings of the EDBT Workshop on Clustering Information over the Web*, pages 547–556, 2004.
- [11] H. Edmundson. New methods in automatic extracting. *Journal of the ACM*, 16(2):264–285, 1969.
- [12] C. D. Fellbaum. *WordNet, an Electronic Lexical Database*. MIT Press, Cambridge MA, 1998.
- [13] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.
- [14] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. In *Technical Report Stanford University*, 1998.
- [15] N. Fuhr, S. Malik, and M. Lalmas. Overview of the initiative for the evaluation of xml retrieval (inex) 2003. In *Proceedings of the Second INEX Workshop*, 2004.
- [16] A. Herschtal and B. Raskutti. Optimising area under the roc curve using gradient descent. In *ICML*, 2004.
- [17] J. Kupiec, J. Pedersen, and F. Chen. A trainable document summarizer. In *Proceedings of the 18th ACM SIGIR Conference*, pages 68–73, 1995.
- [18] D. Lam, S. Rohall, C. Schmandt, and M. Stern. Exploiting e-mail structure to improve summarization. *Technical Report 02-02, IBM Watson Research Centre*, 2002.
- [19] A. M. Lam-Adesina and G. J. F. Jones. Applying summarization techniques for term selection in relevance feedback. In *Proceedings of the 24th ACM SIGIR Conference*, pages 1–9, 2001.
- [20] G. Lebanon and J. Lafferty. Boosting and maximum likelihood for exponential models. *Technical Report CMU-CS-01-144, School of Computer Science, CMU*, 2001.
- [21] K. Litkowski. Text summarization using xml-tagged documents. In *Proceedings of the Document Understanding Conference (DUC'2003)*, pages 58–65, 2003.
- [22] H. P. Luhn. The automatic creation of literature abstracts. *IBM Journal*, 2:159–165, 1958.
- [23] D. Marcu. The automatic construction of large-scale corpora for summarization research. In *Proceedings of the 22nd ACM SIGIR Conference*, pages 137–144, 1999.
- [24] D. Marcu. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, 2000.
- [25] C. Paice. Constructing literature abstracts by computer: techniques and prospects. *Information Processing & Management*, 26(1):171–186, 1990.
- [26] T. Sakai and K. S. Jones. Generic summaries for indexing in information retrieval. In *Proceedings of the 24th ACM SIGIR Conference*, pages 190–198, 2001.
- [27] A. Sengupta, M. Dalkilic, and J. Costello. Semantic thumbnails: a novel method for summarizing document collections. In *Proceedings of the 22nd annual international conference on Design Of Communication*, pages 45–51. ACM Press, 2004.
- [28] SUMMAC. http://www.itl.nist.gov/iaui/894.02/related_projects/tipster_summac/cmp_lg.html.
- [29] M. Symons. Clustering criteria and multivariate normal mixture. *Biometrics*, 37:35–43, 1981.
- [30] S. Teufel and M. Moens. Sentence extraction as a classification task. In *Proceedings of the Intelligent Scalable Text Summarization Workshop, ACL*, pages 58–65, 1997.
- [31] S. Teufel and M. Moens. Summarizing scientific articles – experiments with relevance and rhetorical status. *Computational Linguistics*, 28(4):409–445, 2002.
- [32] A. Tombros and M. Sanderson. Advantages of query biased summaries in information retrieval. In *Proceedings of the 21st ACM SIGIR Conference*, pages 2–10, 1998.
- [33] TreeTagger. <http://www.ims.uni-stuttgart.de/projekte/corplex/treetagger/decisiontreetagger.html>.
- [34] N. Usunier, M. Amini, and P. Gallinari. A data-dependent generalisation error bound for the AUC. In *To appear in the proceedings of the ICML'05 Workshop on ROC Analysis in Machine Learning*, 2005.
- [35] C. Wolf, S. Alpert, J. Vergo, L. Kozakov, and Y. Doganata. Summarizing technical support documents for search: expert and user studies. *IBM Systems Journal*, 43(3):564–586, 2004.
- [36] D. Wolfe and R. Hogg. On constructing statistics and reporting data. *American Statistician*, 25:27–30, 1971.
- [37] J. Xu and W. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th ACM SIGIR Conference*, pages 4–11, 1996.
- [38] L. Yan, R. Dodier, M. Mozer, and R. Wolniewicz. Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistics. In *Proceedings of the 20th International Conference on Machine Learning*, pages 848–855, 2003.