A Classification Algorithm for Predicting the Structure of Summaries

Horacio Saggion
University of Sheffield
211 Portobello Street
Sheffield - S1 4DP
United Kingdom
http://www.dcs.shef.ac.uk/~saggion
H.Saggion@dcs.shef.ac.uk

Abstract

We investigate the problem of generating the structure of short domain independent abstracts. We apply a supervised machine learning approach trained over a set of abstracts collected from abstracting services and automatically annotated with a text analysis tool. We design a set of features for learning inspired from past research in content selection, information ordering, and rhetorical analysis for training an algorithm which then predicts the discourse structure of unseen abstracts. The proposed approach to the problem which combines local and contextual features is able to predict the local structure of the abstracts in just over 60% of the cases.

1 Introduction

Mani (2001) defines an abstract as "a summary at least some of whose material is not present in the input". In a study of professional abstracting, Endres-Niggemeyer (2000) concluded that professional abstractors produce abstracts by "cut-and-paste" operations, and that *standard sentence patterns* are used in their production. Examples of abstracts produced by a professional abstractor are shown in Figures 1 and 2. They contain fragments "copied" from the input documents together with phrases (underlined in the figures) inserted by the professional abstractors. In a recent study in human abstracting (restricted to the amendment of authors abstracts) Montesi and Owen (2007) noted that professional abstractors prepend third person singular verbs in present tense and without subject to the author abstract, a phenomenon related – yet different – from the problem we are investigating in this paper.

Note that the phrases or predicates prepended to the selected *sentence fragments* copied from the input document have a communicative function:

> Presents a model instructional session that was prepared and taught by librarians to introduce college students, faculty, and staff to the Internet by teaching them how to join list-servs and topic- centered discussion groups. Describes the sessions' audience, learning objectives, facility, and course design. Presents a checklist for preparing an Internet instruction session.

Figure 1: Professional Abstracts with Inserted Predicates from LISA Abstracting Service

> Talks about Newsblaster, an experimental software tool that scans and summarizes electronic news stories, developed by Columbia University's Natural Language Processing Group. Reports that Newsblaster is a cross between a search engine and ... Explains that Newsblaster publishes the summaries in a Web page that divides the day summaries into .... Mentions that Newsblaster is considered an aid to those who have to quickly canvas large amounts of information from many sources.

Figure 2: Professional Abstract with Inserted Predicates from Internet & Personal Computing Abstracts

they inform or alert the reader about the content of the abstracted document by explicitly marking what the author *says or mentions*, *presents or introduces*, *concludes*, or *includes*, in her paper. Montesi and Owen (2007) observe that the revision of abstracts is carried out to improve comprehensibility and style and to make the abstract objective.

We investigate how to create the discourse structure of the abstracts: more specifically we are interested in predicting the inserted predicates or phrases and at which positions in the abstract they should be prepended.

Abstractive techniques in text summarization include sentence compression (Cohn and Lapata, 2008), headline generation (Soricut and Marcu, 2007), and canned-based generation (Oakes and Paice, 2001). Close to the problem studied here is Jing and McKeown's (Jing and McKeown, 2000) cut-and-paste method founded on Endres-Niggemeyer's observations. The cut-and-paste

Proceedings of the 2009 Workshop on Language Generation and Summarisation, ACL-IJCNLP 2009, pages 31–38,
Suntec, Singapore, 6 August 2009.

method includes such operations as sentence truncation, aggregation, specialization/generalization, reference adjustment and rewording. None of these operations account for the transformations observed in the abstracts of Figures 1 and 2. The formulaic expressions or predicates inserted in the abstract "glue" together the extracted fragments, thus creating the abstract's discourse structure.

To the best of our knowledge, and with the exception of Saggion and Lapalme (2002) indicative generation approach which included operations to add extra linguistic material to generate an indicative abstract, the work presented here is the first to investigate this relevant operation in the field of text abstracting and to propose a robust computational method for its simulation.

In this paper we are interested in the process of generating the structure of the abstract by automatic means. In order to study this problem, we have collected a corpus of abstracts written by abstractors; we have designed an algorithm for predicting the structure; implemented the algorithm; and evaluated the structure predicted by the automatic system against the true structure.

2 Problem Specification, Data Collection, and Annotation

The abstracts we study in this research follow the pattern:

$$\text{Abstract} \equiv \bigoplus_{i=1}^{n} \text{Pred}_i \oplus \beta_i$$

where Pred_i is a phrase used to introduce the "content" β_i of sentence i, n is the number of sentences in the abstract, $\bigoplus$ indicates multiple concatenation, and $X \oplus Y$ indicates the concatenation of X and Y. In this paper we concentrate only on this "linear" structure, we plan to study more complex (e.g., tree-like representations) in future work.

The problem we are interested in solving is the following: given sentence fragments β_i extracted from the document, how to create the Abstract. Note that if N is the number of different phrases (Pred_i) used in the model, then *a priori* there are N^n possible discourse structures to select from for the abstract, generating all possibilities and selecting the most appropriate would be impractical. We present an algorithm that decides which predicate or phrase is most suitable for each sentence, doing this by considering the sentence content and the abstract generated so far. For the experiments to be reported in this paper, the discourse structure of the abstracts is created using predicates or expressions learned from a corpus a subset of which is shown in Table 1.

We have collected abstracts from various databases including LISA, ERIC, and Internet & Personal Computing Abstracts, using our institutional library's facilities and the abstracts' providers' keyword search facilities. Electronic copies of the abstracted documents can also be accessed through our institution following a link, thus allowing us to check abstracts against abstracted document (additional information on the abstracts is given in the Appendix).

2.1 Document Processing and Annotation

Each electronic version of the abstract was processed using the freely available GATE text analysis software (Cunningham et al., 2002). First each abstract was analyzed by a text structure analysis program to identify meta-data such as title, author, source document, the text of the abstract, etc. Each sentence in the abstract was stripped from the predicate or phrase inserted by the abstractor (e.g., "Mentions that", "Concludes with") and a normalised version of the expression was used to annotate the sentence, in a way similar to the abstracts in Figures 1 and 2. After this each abstract and document title was tokenised, sentence splitted, part-of-speech tagged, and morphologically analyzed. A rule-based system was used to carry out partial, robust syntactic and semantic analysis of the abstracts (Gaizauskas et al., 2005) producing predicate-argument representations where predicates which are used to represent entities are created from the morphological roots of nouns or verbs in the text (unary predicates) and predicates with are used to represent binary relations are a closed set of names representing grammatical relations such as the verb logical object, or the verb logical subject or a prepositional attachment, etc. This predicate-argument structure representation was further analysed in order to extract "semantic" triples which are used in the experiments reported here. Output of this analysis is shown in Figure 3. Note that the representation also contains the tokens of the text, their parts of speech, lemmas, noun phrases, verb phrases, etc.

3 Proposed Solution

Our algorithm (see Algorithm 1) takes as input an ordered list of sentence fragments obtained from the source document and decides how to "paste" the fragments together into an abstract;

to address; to add; to advise; to assert; to claim; to comment; to compare; to conclude; to define; to describe; to discuss; to evaluate; to examine; to explain; to focus; to give; to highlight; to include; to indicate; to note; to observe; to overview; to point out; to present; to recommend; to report; to say; to show; to suggest; ...
to report + to indicate + to note + to declare + to include; to provide + to explain + to indicate + to mention; to point out + to report + to mention + to include; to discuss + to list + to suggest + to conclude; to present + to say + to add + to conclude + to contain; to discuss + to explain + to recommend; to discuss + to cite + to say; ...

Table 1: Subset of predicates or expressions used by professional abstractors and some of the discourse structures used.

Sentence: *Features a listing of ten family-oriented programs, including vendor information, registration fees, and a short review of each.*
Representation: listing-det-a; listing-of-program; family-oriented-adj-program; fee-qual-registration; information-qual-vendor; listing-apposed-information; ...

Figure 3: Sentence Representation (partial)

Algorithm 1 Discourse Structure Prediction Algorithm

Given: a list of n sorted text fragments β_i
begin
Abstract $\leftarrow$ ““;
Context $\leftarrow$ START;
for all $i : 0 \leq i \leq n-1$; **do**
  Pred $\leftarrow$ PredictPredicate(Context, β_i);
  Abstract $\leftarrow$ Abstract $\oplus$ Pred $\oplus$ β_i $\oplus$ “.”;
  Context $\leftarrow$ ExtractContext(Abstract);
end for
return Abstract
end

at each iteration the algorithm selects the “best” available phrase or predicate to prepend to the current fragment from a finite vocabulary (induced from the analysed corpus) based on local and contextual information. One could rely on existing trainable sentence selection (Kupiec et al., 1995) or even phrase selection (Banko et al., 2000) strategies to pick up appropriate β_i's from the document to be abstracted and rely on recent information ordering techniques to sort the β_i fragments (Lapata, 2003). This is the reason why we only address here the discourse structure generation problem.

3.1 Predicting Discourse Structure as Classification

There are various possible ways of predicting what expression to insert at each point in the generation process (i.e., the *PredictPredicate* function in Algorithm 1). In the experiments reported here we use a classification algorithm based on lexical, syntactic, and discursive features, which decides which of the N possible available phrases is most suitable. The algorithm is trained over the annotated abstracts and used to predict the structure of unseen test abstracts.

Where the classification algorithm is concerned, we have decided to use Support Vector Machines which have recently been used in different tasks in natural language processing, they have been shown particularly suitable for text categorization (Joachims, 1998). We have tried other machine learning algorithms such as Decision Trees, Naive Bayes Classification, and Nearest Neighbor from the Weka toolkit (Witten and Frank, 1999), but the support vector machines gave us the best classification accuracy (a comparison with Naive Bayes will be presented in Section 4).

The features used for the experiments reported here are inspired by previous work in text summarization on content selection (Kupiec et al., 1995), rhetorical classification (Teufel and Moens, 2002), and information ordering (Lapata, 2003). The features are extracted from the analyzed abstracts with specialized programs. In particular we use positional features (position of the predicate to be generated in the structure), length features (number of words in the sentence), title features (e.g., presence of title words in sentence), content features computed as the syntactic head of noun and verb phrases, semantic features computed as the

to add; to conclude; to contain; to describe; to discuss; to explain; to feature; to include; to indicate; to mention; to note; to point out; to present; to provide; to report; to say

Table 2: Predicates in the reduced corpus

arguments of "semantic" triples (Section 2.1) extracted from the parsed abstracts. Features occurring less than 4 times in the corpus were removed for the experiments. For each sentence, a cohesion feature is also computed as the number of nouns in common with the previous sentence fragment (or title if first sentence). Cohesion information has been used in rhetorical-based parsing for summarization (Marcu, 1997) in order to decide between "list" or "elaboration" relations and also in content selection for summarization (Barzilay and Elhadad, 1997). For some experiments we also use word-level information (lemmas) and part-of-speech tags. For some of the experiments reported here the variable *Context* at iteration i in Algorithm 1 is instantiated with the predicates predicted at iterations $i-1$ and $i-2$.

4 Experiments and Results

The experiments reported here correspond to the use of different features as input for the classifier. In these experiments we have used a subset of the collected abstracts, they contain predicates which appeared at least 5 times in the corpus. With this restriction in place the original set of predicates used to create the discourse structure is reduced to sixteen (See Table 2), however, the number of possible structures in the reduced corpus is still considerable with a total of 179 different structures.

In the experiments we compare several classifiers:

- Random Generation selects a predicate at random at each iteration of the algorithm;
- Predicate-based Generation is a SVM classifier which uses the two previous predicates to generate the current predicate ignoring sentence content;
- Position-based Generation is a SVM classifier which also ignores sentence content but uses as features for classification the absolute position of the sentence to be generated;

Configuration	**Avg.Acc**
Random Generation	10%
Predicate-based Generation	35%
Position-based Generation	38%
tf*idf-based Generation	55%
Summarization-based Generation	60%

Table 3: Average accuracy of different classification configurations.

- *tf*idf*-based Generation is a SVM classifier which uses lemmas of the sentence fragment to be generated to pick up one predicate (note that position features and predicates were added to the mix without improving the classifier);
- Summarization-based Generation is a SVM which uses the summarization and discourse features discussed in the previous section including contextual information ($Pred_{i-2}$ and $Pred_{i-1}$ – with special values when $i = 0$ and $i = 1$).

We measure the performance of each instance of the algorithm by comparing the predicted structure against the true structure. We compute two metrics: (i) accuracy at the sentence level (as in classification), which is the proportion of predicates which were correctly generated; and (ii) accuracy at the textual level, which is the proportion of abstracts correctly generated. For the latter we compute the proportion of abstracts with zero errors, less than two errors, and less than three errors.

For every instance of the algorithm we perform a cross-validation experiment, selecting for each experiment 20 abstracts for testing and the rest of the abstracts for training. Accuracy measures at sentence and text levels are averages of the cross-validation experiments.

Results of the algorithms are presented in Tables 3 and 4. Random generation has very poor performance with only 10% local accuracy and less than 1% of full correct structures. Knowledge of the predicates selected for previous sentences improves performance over the random system (35% local accuracy and 5% of full correct structures predicted). As in previous summarization studies, position proved to contribute to the task: the positional classifier predicts individual predicates with a 38% accuracy; however only 8% of

the structures are recalled. Differences between the accuracies of the two algorithms (predicate-based and position-based) are significant at 95% confidence level (a *t-test* was used). As it is usually the case in text classification experiments, the use of word level information (lemmas in our case) achieves good performance: 55% classification accuracy at sentence level, and 18% of full structures correctly predicted. The use of lexical (noun and verb heads, arguments), syntactic (parts of speech information), and discourse (predicted predicates, position, cohesion) features has the better performance with 60% classification accuracy at sentence level predicting 21% of all structures with 73% of the structures containing less than 3 errors. The differences in accuracy between the word-based classifier and the summarization-based classifier are statistically significant at 95% confidence level (a *t-test* was used). A Naive Bayes classifier which uses the summarization features achieves 50% classification accuracy.

Conf.	**0 errs**	<2 **errrs**	<3 **errs**
Random	0.3%	4%	20%
Predicate-based	5%	24%	48%
Position-based	8%	33%	50%
tf*idf-based	18%	42%	67%
Summ-based	21%	55%	73%

Table 4: Percent of correct and partially correct structures predicted. Averaged over all runs.

Table 5 shows a partial confusion table for predicates "to add", "to conclude", "to explain", and "to present" while and Table 6 reports individual classification accuracy. All these results are based on averages of the summarization-based classifier.

5 Discussion

We have presented here a problem which has not been investigated before in the field of text summarization: the addition of extra linguistic material (i.e., not present in the source document) to the abstract "informational content" in order to create the structure of the abstract. We have proposed an algorithm which uses a classification component at each iteration to predict predicates or phrases to be prepended to fragments extracted from a document. We have shown that this classifier based on summarization features including linguistic, semantic, positional, cohesive, and discursive information can predict the local discourse structures in over 60% of the cases. There is a mixed picture on the prediction of individual predicates, with most predicates correctly classified in most of the cases except for predicates such as "to describe", "to note", and "to report" which are confused with other phrases. Predicates such as "to present" and "to include" have the tendency of appearing towards the very beginning or the very end of the abstract been therefore predicted by position-based features (Edmundson, 1969; Lin and Hovy, 1997). Note that in this work we have decided to evaluate the predicted structure against the true structure (a hard evaluation measure), in future work we will assess the abstracts with a set of quality questions similar to those put forward by the Document Understanding Conference Evaluations (also in a way similar to (Kan and McKeown, 2002) who evaluated their abstracts in a retrieval environment). We expect to obtain a reasonable evaluation result given that it appears that some of the predicates or phrases are "interchangeable" (e.g., "to contain" and "to include").

Actual Pred.	**Predicted Pred.**	**Conf.Freq.**
to add	to add	32%
	to explain	16%
	to say	10%
to conclude	to conclude	35%
	to say	29%
	to add	7%
to explain	to explain	35%
	to say	15%
	to add	11%
to present	to present	86%
	to discuss	7%
	to provide	1%

Table 5: Classification Confusion Table for a Subset of Predicates in the Corpus (Average Frequency).

6 Related Work

Liddy (1991) produced a formal model of the informational or conceptual structure of abstracts of empirical research. This structure was elicited from abstractors of two organizations ERIC and PsycINFO through a series of tasks. Lexical clues which predict the components of the structure were latter induced by corpus analysis. In the domain of indicative summarization, Kan and McK-

Predicate	Avg. Accuracy
to add	31.40
to conclude	34.78
to contain	10.96
to describe	15.69
to discuss	54.55
to explain	35.63
to feature	34.38
to include	85.86
to indicate	20.69
to mention	26.47
to note	6.78
to point out	91.67
to present	86.19
to provide	40.94
to report	1.59
to say	75.86

Table 6: Predicate Classification Accuracy

eown (2002) studied the problem of generating abstracts for bibliographical data which although in a restricted domain has some contact points with the work described here. As in their work we use the abstracts in our corpus to induce the model. They rely on a more or less fixed discourse structure to accommodate the generation process. In our approach the discourse structure is not fixed but predicted for each particular abstract. Related to our classification experiments is work on semantic or rhetorical classification of "structured" abstracts (Saggion, 2008) from the MEDLINE abstracting database where similar features to those presented here were used to identify in abstracts semantic categories such as objective, method, results, and conclusions. Related to this is the work by Teufel and Moens (2002) on rhetorical classification for content selection. In cut-and-paste summarization (Jing and McKeown, 2000), sentence combination operations were implemented manually following the study of a set of professionally written abstracts; however the particular "pasting" operation presented here was not implemented. Previous studies on text-to-text abstracting (Banko et al., 2000; Knight and Marcu, 2000) have studied problems such as sentence compression and sentence combination but not the "pasting" procedure presented here. The insertion in the abstract of linguistic material not present in the input document has been addressed in paraphrase generation (Barzilay and Lee, 2004) and canned-based summarization (Oakes and Paice, 2001) in limited domains. Saggion and Lapalme (2002) have studied and implemented a rule-based "verb selection" operation in their SumUM system which has been applied to introduce document topics during indicative summary generation.

Our discourse structure generation procedure is in principle generic but depends on the availability of a corpus for training.

7 Conclusions

In text summarization research, most attention has been paid to the problem of what information to select for a summary. Here, we have focused on the problem of how to combine the selected content with extra linguistic information in order to create the structure of the summary.

There are several contributions of this work:

- First, we have presented the problem of generating the discourse structures of an abstract and proposed a meta algorithm for predicting it. This problem has not been investigated before.
- Second, we have proposed – based on previous summarization research – a number of features to be used for solving this problem; and
- Finally, we have propose several instantiations of the algorithm to solve the problem and achieved a reasonable accuracy using the designed features;

There is however much space for improvement even though the algorithm recalls some "partial structures", many "full structures" can not be generated. We are currently investigating the use of induced rules to address the problem and will compare a rule-based approach with our classifier. Less superficial cohesion features are being investigated and will be tested in this classification framework.

Acknowledgements

We would like to thank three anonymous reviewers for their suggestions and comments. We thank Adam Funk who helped us improve the quality of our paper. Part of this research was carried out while the author was working for the EU-funded MUSING project (IST-2004-027097).

References

Michele Banko, Vibhu O. Mittal, and Michael J. Witbrock. 2000. Headline generation based on statistical translation. In *ACL '00: Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 318–325, Morristown, NJ, USA. Association for Computational Linguistics.

Regina Barzilay and Michael Elhadad. 1997. Using Lexical Chains for Text Summarization. In *Proceedings of the ACL/EACL'97 Workshop on Intelligent Scalable Text Summarization*, pages 10–17, Madrid, Spain, July.

R. Barzilay and L. Lee. 2004. Catching the Drift: Probabilistic Content Models, with Applications to Generation and Summarization. In *Proceedings of HLT-NAACL 2004*.

T. Cohn and M. Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of COLING 2008*, Manchester.

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: A framework and graphical development environment for robust NLP tools and applications. In *ACL 2002*.

H.P. Edmundson. 1969. New Methods in Automatic Extracting. *Journal of the Association for Computing Machinery*, 16(2):264–285, April.

Brigitte Endres-Niggemeyer. 2000. SimSum: an empirically founded simulation of summarizing. *Information Processing & Management*, 36:659–682.

R. Gaizauskas, M. Hepple, H. Saggion, and M. Greenwood. 2005. SUPPLE: A Practical Parser for Natural Language Engineering Applications.

Hongyan Jing and Kathleen McKeown. 2000. Cut and Paste Based Text Summarization. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 178–185, Seattle, Washington, USA, April 29 - May 4.

T. Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning (ECML)*, pages 137–142, Berlin. Springer.

Min-Yen Kan and Kathleen R.. McKeown. 2002. Corpus-trained text generation for summarization. In *Proceedings of the Second International Natural Language Generation Conference (INLG 2002)*, pages 1–8, Harriman, New York, USA.

Kevin Knight and Daniel Marcu. 2000. Statistics-based summarization - step one: Sentence compression. In *Proceedings of the 17th National Conference of the American Association for Artificial Intelligence*. AAAI, July 30 - August 3.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A Trainable Document Summarizer. In *Proc. of the 18th ACM-SIGIR Conference*, pages 68–73, Seattle, Washington, United States.

M. Lapata. 2003. Probabilistic Text Structuring: Experiments with Sentence Ordering. In *Proceedings of the 41st Meeting of the Association of Computational Linguistics*, pages 545–552, Sapporo, Japan.

Elizabeth D. Liddy. 1991. The Discourse-Level Structure of Empirical Abstracts: An Exploratory Study. *Information Processing & Management*, 27(1):55–81.

C. Lin and E. Hovy. 1997. Identifying Topics by Position. In *Fifth Conference on Applied Natural Language Processing*, pages 283–290. Association for Computational Linguistics, 31 March-3 April.

Inderjeet Mani. 2001. *Automatic Text Summarization*. John Benjamins Publishing Company.

D. Marcu. 1997. *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts*. Ph.D. thesis, Department of Computer Science, University of Toronto.

M. Montesi and J. M. Owen. 2007. Revision of author abstracts: how it is carried out by LISA editors. *Aslib Proceedings*, 59(1):26–45.

M.P. Oakes and C.D. Paice. 2001. Term extraction for automatic abstracting. In D. Bourigault, C. Jacquemin, and M-C. L'Homme, editors, *Recent Advances in Computational Terminology*, volume 2 of *Natural Language Processing*, chapter 17, pages 353–370. John Benjamins Publishing Company.

H. Saggion and G. Lapalme. 2002. Generating Indicative-Informative Summaries with SumUM. *Computational Linguistics*, 28(4):497–526.

H. Saggion. 2008. Automatic Summarization: An Overview. *Revue Française de Linguistique Appliquée* , XIII(1), Juin.

R. Soricut and D. Marcu. 2007. Abstractive headline generation using WIDL-expressions. *Inf. Process. Manage.*, 43(6):1536–1548.

S. Teufel and M. Moens. 2002. Summarizing Scientific Articles: Experiments with Relevance and Rhetorical Status. *Computational Linguistics*, 28(4):409–445.

Ian H. Witten and Eibe Frank. 1999. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, October.

Appendix I: Corpus Statistics and Examples

The corpus of abstracts following the specification given in Section 2 contains 693 abstracts, 10,423

sentences, and 305,105 tokens. The reduced corpus used for the experiments contains 300 abstracts.

Examples

Here we list one example of the use of each of the predicates in the reduced set of 300 abstracts used for the experiments.

> Adds that it uses search commands and features that are similar to those of traditional online commercial database services, has the ability to do nested Boolean queries as well as truncation when needed, and provides detailed documentation that offers plenty of examples.
>
> Concludes CNET is a network of sites, each dealing with a specialized aspect of computers that are accessible from the home page and elsewhere around the site.
>
> Contains a step-by-step guide to using PGP.
>
> Describes smallbizNet, the LEXIS-NEXIS Small Business Service, Small Business Administration, Small Business Advancement National Center, and other small business-related sites.
>
> Discusses connections and links between differing electronic mail systems.
>
> Explains DataStar was one of the first online hosts to offer a Web interface, and was upgraded in 1997.
>
> Features tables showing the number of relevant, non-relevant, and use retrievals on both LEXIS and WIN for federal and for state court queries.
>
> Includes an electronic organizer, an ergonomically correct keyboard, an online idle-disconnect, a video capture device, a color photo scanner, a real-time Web audio player, laptop speakers, a personal information manager (PIM), a mouse with built-in scrolling, and a voice fax-modem.
>
> Indicates that the University of California, Berkeley, has the School of Information Management and Systems, the University of Washington has the Information School, and the University of Maryland has the College of Information Studies.
>
> Mentions that overall, the interface is effective because the menus and sear screens permit very precise searches with no knowledge of searching or Dialog databases.
>
> Notes that Magazine Index was originally offered on Lyle Priest's invention, a unique microfilm reader.
>
> Points out the strong competition that the Internet has created for the traditional online information services, and the move of these services to the Internet.
>
> Presents searching tips and techniques.
>
> Provides summaries of African art; Allen Memorial Art Museum of Oberlin College; Art crimes; Asian arts; Da Vinci, Leonardo; Gallery Walk; and Native American Art Gallery.
>
> Reports that Dialog has announced major enhancements to its alerting system on the DialogClassic, DialogClassic Web, and DialogWeb services.
>
> Says that dads from all over the country share advice on raising children, educational resources, kids' software, and other related topics using their favorite online service provider.