Corpus-trained text generation for summarization

Min-Yen Kan and **Kathleen R. McKeown**
Department of Computer Science
Columbia University
New York, NY 10027, USA
{min,kathy}@cs.columbia.edu

Abstract

We explore how machine learning can be employed to learn rulesets for the traditional modules of content planning and surface realization. Our approach takes advantage of semantically annotated corpora to induce preferences for content planning and constraints on realizations of these plans. We applied this methodology to an annotated corpus of indicative summaries to derive constraint rules that can assist in generating summaries for new, unseen material.

1 Introduction

Traditional natural language generation (NLG) approaches rely heavily on human experts to code discourse, semantic, and lexical resources. These resources are used by systems to determine the discourse and sentential structure of the text, and its word choice. This process can be very time consuming, involving experts that examine target documents and distill proper discourse plans and lexicons that can produce the desired text.

In this paper, we investigate a novel approach which automatically acquires such knowledge using an annotated training corpus. Our method constructs summarization system components by first learning high-level content planning patterns and then learning low-level constraints on how to realize these content plans in natural language. By applying this approach to a training corpus consisting of documents belonging to the same domain and genre, the system can generate a model for production of similar texts. We show how this framework can be applied to automatic text summarization by using a corpus of annotated bibliography entries as the training corpus to produce a model of indicative summaries (Cremmins, 1982). These entries discuss different books but express the same reoccurring types of information using different surface forms.

While the corpus from which plans and realization patterns are acquired is restricted to input documents of the same genre that exhibit structural regularity, the learned plans can be applied to other domains and genres. In this paper, we draw on input from the genre of annotated bibliography entries, but will apply the learned plans to generate summaries of web-available consumer health texts.

A content plan consists of *predicates* specifying what kind of information should occur in what order in a generated summary. Each predicate will ultimately be realized by one of the lexicalized phrases that are associated with it. The research we present focuses on learning rules that can predict the order of predicates in a text and acquiring the lexicalized phrases associated with each predicate, and is illustrated in Figure 1.

The acquisition of the content planning ruleset works by finding occurrence patterns of predicates in manually annotated training corpora. This module determines what predicates are required or optional in the plan, and uncovers ordering constraints between them. Our approach in acquiring content planning rules differs from related work in its integration of contextual constraints.

A second acquisition component for partial surface realization considers frequent lexical dependency patterns that are unique to specific predicates (e.g., the *Audience* predicate in bibliography entries)

as predicate realizations and uncovers constraints governing their usage. These patterns distinguish between constituents that determine the semantics of a predicate (which we call a predicate's *attributes*) as well as other *associated text* constituents that are used to convey the information (e.g., surrounding common phrases) in different surface forms.

In this paper, we first describe the role of indicative summaries and show how their generation can be viewed as an instance of our task. We then explain how our two acquisition algorithms function, drawing examples from indicative summary generation. We examine the acquisition process for content planning first, and partial surface realization second. We show how these learned constraints can be applied to generate new summaries in the conclusion.

2 Application to summarization

Automatic Text Summarization (ATS) is the process of using a computerized algorithm to condense documents into a shorter form (for a current overview, see (Mani and Maybury, 1999)). A particular type of document summary that is the focus of our studies is the *indicative summary*, a type of summary that hints at a document's content and does not substitute for the full text. Card catalog entries from a library catalog and annotated bibliography entries are examples of this type, and typically summarize a book in the span of a few sentences. Such texts fall within a single genre and thus fulfill our input prerequisite. We have applied our corpus-trained technique to a corpus of annotated bibliography entries and learned what kinds of content (i.e., predicates) are included and their ordering (the content planning module), as well as learned how these predicates are expressed (the partial surface realization module).

In a generation phase not detailed here, these trained modules will produce multidocument summaries for sets of consumer health texts that vary greatly in discourse structure, length, topic and wording (Kan et al., 2001b). The learned plans are used to determine how to present these indicative differences using text generation, in contrast to other systems that use sentence extraction. Unlike other generation systems that generate text from semantic input, our summarization system uses the plans to select content from full text and to generate variability in syntax and phrasing by choosing wordings from variants of full phrases.

To investigate the viability of producing indicative summaries using this approach, we collected a corpus consisting of 2000 bibliography entries that have been collected from various websites over various domains of knowledge. We processed the corpus with Collins' lexical dependency based parser (Collins, 1996), and also added word stem information using the Porter algorithm.

3 Semantic annotation of summary corpora

Automatic semantic tagging of the corpus allows us to infer what predicates are typically included in indicative summaries. In our corpus of 2000 summaries, we annotated a random 5% (= 100) of the entries. We used the decision tree learner, `ripper` (Cohen, 1995), to induce a decision tree that was used to automatically label a new corpus with predicates, and used 5-fold cross validation to ensure results were stable. We expanded on our previous indicative summary tagset from (Kan et al., 2001a) to a total of 24 predicates, detailed in Tables 1 and 2.

Nodes in the parse trees (corresponding to sentences, phrases or individual words) in the training portion of the corpus were tagged by one of the authors. Automatic tagging thus assigns one of these 25 predicates (the 24 plus a default "none") to each node in the parse tree. By default, tagging all nodes with "none" gives a high baseline accuracy of 99.47% (all 15,208 parse nodes in the 100 entries), but 0% accuracy on the 24 semantic predicates. This was improved to 66% accuracy, as shown in Table 3 by using features that represent the predicate's set of words, and relative and absolute position in the summary. We further introduced the features that model local context of the preceeding and succeeding predicates, and features that model language genericity which marginally improved performance. The genericity feature captures how uniform the language is for particular predicates across instances. The idea was that the topicality predicates (in Table 1) that express domain-specific knowledge would vary in vocabulary across instances, but that metadata predicates (in Table 2, such as *Audience*) would have a more stable vocabulary.

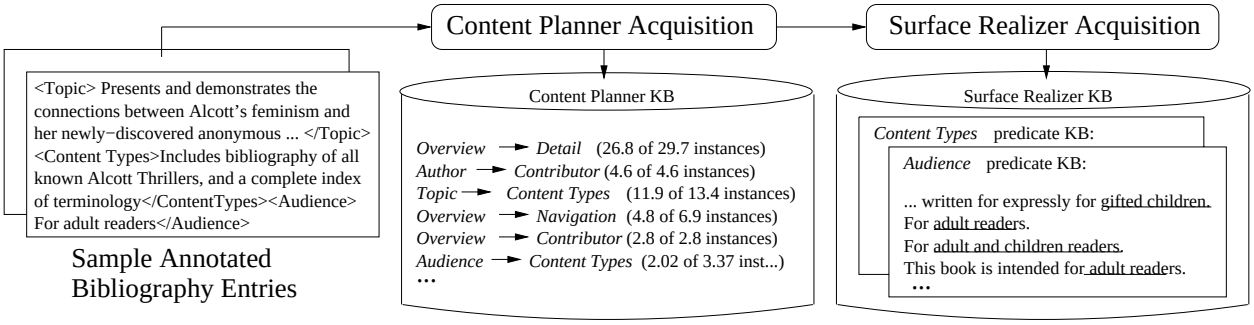

Figure 1: The content plan is a collection of probabilistic ordering constraints, while the surface realizer consists of attribute values (underlined), and associated text that convey the predicate's semantics.

Predicate	# of occurrences	% entries having predicate
Detail	139	47%
Quotations, extracted sentences, parts of a chronology, conclusions		
Overview	72	64%
(Generalized description of the entire resource, "This book is about Louisa Alcott's life.")		
Topic	34	28%
(High-level **list** of topics, e.g., "Topics include symptoms, treatment ...")		

Table 1: Distribution of content-based topicality predicates in the 100-entry annotated corpus.

Further analysis reveals that certain predicates are recovered more often than others. For example, topicality predicates occur with less regularity and display more variability in their expression and thus are more difficult to recover. Tags that occur seldomly are also not recovered by the current set of features because of data sparseness. We feel that an expansion of the fully annotated corpus or additional annotation with respect to these more sparse tags would improve performance here.

4 Learning for the content planner

The semantically annotated corpus is the basis for learning the rule base for content planning. These rules determine what the text and discourse structure should look like, both in terms of a) content ("what to say") and b) its ordering ("where to say it"). We examine each of these two tasks in turn.

Content determination. Documents in our indicative summary corpus discuss different books and thus have different predicate attribute values. In addition, some predicates are present in some summaries and not in others (e.g., *author* or *editor*). Tables 1 and 2 list the predicates and their frequency in the training corpus. The presence of the predicate may also be dependent on its value (e.g., *Edition* only occurs after the first edition).

Content ordering. The presence or absence of particular predicates depends greatly on the presence or absence of its peers. Thus it is important to encode content structuring information, represented as local preferences rather than predefined schemas. Duboue and McKeown (2001) detail an approach for this problem which we initially tried that uses techniques from computational biology, but which is best suited for summaries with multiple instances of the same predicate. Instead, we calculated bigram statistics on pairs of adjacent predicates, recording which occurred before another. These statistics are used to find an ordering of the predicates that maximizes agreement with training observations. This approach was also utilized in work done on premodifier ordering (Shaw and Hatzivassiloglou, 1999), in which pairs of premodifiers were observed and used to find ordering constraints. The technique is also referred to as Majority Ordering in (Barzilay et al., 2001), in which bigram orderings were elicited from human subjects.

(*Background* | *Language*) *Overview Topic Size Media_Types Authority Collection_Size* (*Comparison* | *Detail* | *Content_Types* | *Navigation* | *Query_Relevance*) *Subjective Difficulty Author Purpose Style* (*Publisher* | *Award* | *Readability* | *Audience* | *Contributor* | *Copyright*)

Figure 2: Highest agreement full orderings of the predicates using harmonic penalties. Predicates are swappable where "|" occurs.

We augmented the basic approach by expand-

Predicate	# of occurrences	% entries having predicate
Media Type	55	48%
(e.g. "This book ...", "A weblet ...", "Spans 2 CDROMs")		
Author / Editor	43	27%
Content Types	41	29%
(e.g. "figures and tables")		
Subjective Assessment	36	24%
(e.g. "highly recommended")		
Authority / Authoritativeness	26	20%
Background / Source	21	16%
(e.g. "based on a report")		
Navigation / Internal Structure	16	11%
(e.g. "is organized into three parts")		
Collection Size	13	10%
Purpose	13	10%
Audience	12	12%
(e.g. "for adult readers")		
Contributor	12	12%
Name of the author of the annotated entry		
Cross-resource Comparison	10	9%
(e.g., "similar to the other articles"		
Size/Length	9	7%
Style	8	6%
(e.g., "in verse rhythm", "showcased in soft watercolors")		
Query Relevance	4	3%
(text relevant to the theme of the collection)		
Readability	4	4%
Difficulty	4	4%
(e.g., "requires no matrix algebra")		
Edition / Publication	3	3%
Language	2	2%
Copyright	2	1%
Award	2	1%

Table 2: Distribution of metadata and document-derivable predicates in the 100-entry corpus.

ing the statistic to account for longer distance co-occurences. Our statistic better models the fading strength of context farther from the decision point by utilizing information provided in all previous n predicates. We constructed two backoff schemes: one based on the harmonic series, the other based on the quadratic. In both, a precedence relationship of distance one (e.g. adjacent) is given a full strength score, but a distance n relationship is given $\frac{1}{n}$ unit score in the harmonic and $\frac{1}{2^n}$ in the quadratic. Each particular pair of different predicates acculmulate these weights as instances are found in the training corpus, and a randomized hill-climbing algorithm is used to find a maximally compliant ordering.

We use both the content determination and content ordering algorithms to generate a new summary discourse plan. To do this, we examine which predicates are found in the library cataloguing database. We use the content determination probabilities to pick m number of predicates to be realized, where m is the user-defined desired summary length. A randomized algorithm selects n predicates from both topicality (multiple selection allowed) and metadata (selectable once only) catagories, biased for the percentages shown in Tables 1 and 2. The predicates are ordered using either the harmonic or quadratic penalized version of the algorithm and result in a discourse plan for the summary.

Accuracy	Feature Type			
	majority baseline	+ lexical	+ parse node & positional	+ contextual & genericity
24 predicates	0%	9%	66%	67%
24 + "none"	99.47%	99.51%	99.82%	99.83%

Table 3: Summary semantic annotation accuracy, using 5-fold C.V. Features are cumulative l to r.

5 Learning for partial surface realization

While content planning concerns itself with the presence or absence of predicates and their ordering, the task of surface realization is to convey the predicates as natural language. In traditional NLG, surface realization is often broken down into three separate tasks: (1) sentence planning, which takes individual messages or propositions and assigns them to specific sentences and determines the sentences' basic syntactic structure; (2) lexical choice, which determines the words used, and (3) syntactic realization, which uses a grammar to produce the sentence. We are concerned with tasks 1 and 2. While there has been work concentrating on inducing syntactic generators (Langkilde, 2000; Bangalore and Rambow, 2000; Ratnaparkhi, 2000; Varges and Mellish, 2001), for specific domains and general language, there has been less work on other generation components (Oh and Rudnicky, 2000).

Certain predicates, such as those that are content- or topic-based (e.g., *Overview* and *Detail* features), which are highly specific to the resource being summarized are best handled by existing techniques of sentence extraction or domain- and genre-specific text grammars (Liddy, 1991; Rama and Srinivasan, 1993). However, many other predicates are more domain-independent (e.g., *Content Types* and *Audi-*

ence). We focus on these metadata predicates, as they comprise a large portion (57%) of the entries.

In our framework, a predicate has two components (also shown in right hand side of Figure 1): the *attribute value* itself ("adult readers") and the *associated text* that is used to cast this information in the semantic role dictated by the predicate ("This book is meant for <attribute_value>" for the *Audience* predicate)[1]. In a stemmed dependency framework, the attribute value is the child and the associated text the head of a dependency relationship (e.g., stemmed: "this book be mean for"$_{head}$ $\rightarrow$ "adult reader"$_{child}$). In this framework, surface realization begins with the process of choosing the most appropriate associated text among alternatives found in the training text, given input attribute values. The associated text and attributes are then realized as sentences, phrases or words, which are combined to form a new text by a sentence planner.

The first task is to differentiate attribute values from the associated text in the training corpus. Our starting point is the collection of sentences or phrases in the annotated corpus that are instances of the same predicate (e.g., a collection of *Audience* sentences). Our analysis of these texts indicates that attribute values are highly flexible in location within the texts and in grammatical structure. In order to encode this flexibility, we capitalize on the stemmed, lexical dependency framework used in parsing the entries. The framework conflates phrases such as "index included", "includes an index" and "inclusion of indices" (found in instances of the *Content Types* predicate) together into a single stemmed lexical dependency pair of "include"$_{head}$ $\rightarrow$ "index"$_{child}$. For each collection of predicate instances, our strategy first identifies highly frequent (threshold = $\bar{x} + 2\sigma_x$) stemmed lexical dependency pairs. Frequent child lexical items in the dependency pair are potential attribute values in the sentence ("index"$_{child}$ as an attribute value for *Content Types*, as seen in Figure 3, #3). From this set, we remove frequent dependency pairs that occur with other predicates; this prevents frequent, corpus-wide dependencies such as "book"$_{head}$ $\rightarrow$ "this"$_{child}$ from appearing as potential attribute values, as they are not exclusively frequent within a single predicate (so #1 and #2 are not attribute values for any of the 24 semantic predicates).

#1, *Topicality*, book$_{head}$ $\rightarrow$ this$_{child}$:
(e.g., "This book discusses Alcott's works ...", "this book covers the theories")
#2, *Content Types*, book$_{head}$ $\rightarrow$ this$_{child}$:
(e.g., "This book also comes with a biography", "is discussed in this book")
#3, *Content Types*, include$_{head}$ $\rightarrow$ index$_{child}$:
(e.g.,"Indices are included", "includes an index", "The book includes an index")
...
[below threshold]
#30, *Content Types*, include$_{head}$ $\rightarrow$ figure$_{child}$:
(e.g., "Includes figures", "figures and tables are included")
...

Figure 3: A portion of the list of stemmed lexical dependencies for various predicates, sorted by frequency. Source snippets are given after the dependency pair.

This method gave good (94.8%) precision, but poor recall, due to the high threshold. To increase recall, we noted that heads of these frequent dependency pairs also served as heads in dependencies with other less frequent child words (dependency head "include" supports the frequent attribute value "index" in #3, but also less frequent one such as "figure" in #30). Including such pairs recovered these less frequent attribute values (95% additional attributes were recovered) with a minimal increase in error (92.2% precision). Thus, in the simplified example in Figure 3, "index" and "figure" are attribute values for the *Content Types* predicate.

Text not identified as attribute values are labeled *associated text*. From our perspective, these lexical dependencies embody lexical choice and resulting syntactic choices internal to the predicate: the alternative forms of the associated texts help to convey the same semantic information (the attribute value) but with different words and syntactic structures. Which alternative is selected is subject to a number of parameters, including stylistic ones. For the surface realizer to make these decisions thus requires that we learn stylistic constraints and apply them in the generation process. We break this process down into three subtasks:

- **Identify and encode linguistic features**. We examined several linguistic features for their potential to predict which alternative associated text are used in particular contexts. Related

[1]Our notion of "predicate" is identical to (Varges and Mellish, 2001) notion of "slots" or "tags"; similarly, our "attribute values" are equivalent to their term "fillers".

work on descriptive appositive language reuse (Radev, 1998), and genre identification (Biber, 1989; Karlgren and Cutting, 1994; Kessler et al., 1997) defines a large set of basic features to use such as character and word-level features, and positional and contextual features. We implemented a total of 27 of these features in our work to test their efficacy in identifying appropriate constraints.

Our problem is related to these previous studies, but differs in some key respects. Choosing descriptions often involves choosing between descriptions that convey different semantic information ("Clinton" as "senator" versus "president"), whereas our associated texts generally convey the same information but realize it differently, similar to paraphrasing (Barzilay and McKeown, 2001). Genre identification differs from our problem mostly in scale; whereas whole texts are input to the genre categorization process, in our problem we have access mostly to single sentences or clauses.

Genre identification work primarily focuses on surface level features rather than assuming a full parse of the text. Since we have access to a full parse, we also model features that we believe have a stylistic impact. We introduce an additional 8 features that look for different types of adjunct constructions, relative clause construction, and passive constructions.

- **Use machine learning to predict linguistic features for the target predicate**. We employed the decision tree learner, `ripper`, to determine which features play a role in predicting the characteristics of the target associated text. We compute values of all 35 (27 + 8 new) features for four different contexts surrounding the target predicate: the (1) previous and (2) next predicates, as well as (3) global and (4) composite features that combine other basic features (e.g., difference between previous and next sentence length). This total of 4 x 35 = 140 features, are used to predict the values of the same features in the target predicate. Since there are a total of 35 features, this results in 35 different machine learners that separately predict one of the 35 features of the target predicate, shown graphically in Figure 4.

  Table 4 counts the occurrences of these 35 features in the induced `ripper` ruleset that is used to structure the target predicate. This measure can be used to assess their relative importance. These results show that our additional features are useful in modeling stylistics but that the literature contributes significant features as well.

2 features from (Radev98)	**20 features from (Karlgren& Cutting94)**	**5 char-level cues from (KesslerE-tAl97)**	**8 new features**
20	25	7	24

Table 4: The number of features used by `ripper` to determine the output feature of target predicates.

- **Find best possible match between predicted features and candidate predicate**. The final step is to use the predicted features of the predicate to match a most appropriate associated text to convey the predicate.

  If the predicted features exactly match a training example's associated text then the selection is trivial. However in practice this rarely occurs and we must select from the available associated texts. As the choice is limited to whole associated texts and not constituents as in other stochastic approaches (Langkilde, 2000; Varges and Mellish, 2001), this search process is constrained and does not present an efficiency problem. For numerical features (e.g., number of words), we use a normalized difference between the desired value and available values from the training associated text to calculate its goodness of fit. For set valued features (e.g. parse node type: NP versus PP), the feature either matches or does not (1 or 0).

  Our algorithm weighs all features equally, and an associated text is chosen such that the matching score is maximized. In future work, we will use human judgments of the realized predicates in context to induce more appropriate feature weights, and will evaluate this module's efficacy for generating appropriate text.

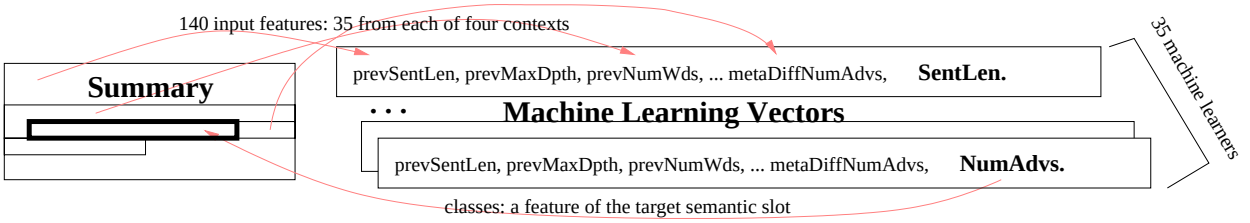

Figure 4: Machine learning architecture for the features of the target predicate, with sample tuples shown.

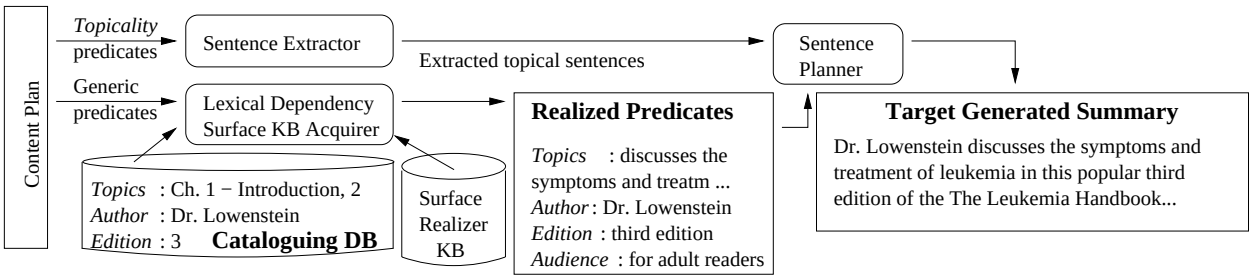

Figure 5: Post content planner architecture with summary

6 Using the rulesets for generation

This approach in this paper constructs rulesets that capture patterns at both the content planning and surface realization levels. While the work here does not constitute an end-to-end generation system, the algorithms for learning these knowledge bases are implemented, and can be applied along with a source of predicate attributes to generate new texts. In our application of indicative summarization generation, cataloging records (such as the U.S. Machine Readable Cataloging (MARC) guidelines (Library of Congress, 2000)) can provide these predicate attributes. Attribute values from the resource's cataloguing record would interact with the surface realization constraints to produce a set of sentences or phrases that correspond to each predicate. A sentence planning module could take the realized fragments and organize them into sentences (according to the content plan) to form a final generated text. Figure 5 shows how this portion of the process would work in conjunction with sentence extraction to find sentences for any topicality predicates.

7 Conclusion

In previous work, we performed a task-based evaluation of a rule-based indicative summary generation system. The study suggested that users were more satisfied with this approach to presenting information retrieval results over other visualization approaches (Kan and Klavans, 2002). The current system described in this paper improves the system with additional flexibility and variability in generation that is a key characteristic of human-produced natural language. Task-based evaluation of this current generation system is currently being done to assess its effectiveness.

In this paper, we have described a new architecture for NLG that takes advantage of annotated corpora. Our method takes a new approach to NLG by using machine learning to capture semantic and stylistic constraints that are traditionally hand coded by human experts. The induced constraints can be used by traditional content planning and surface realization in making their decisions.

To the best of our knowledge, our approach is the first to use lexical dependencies in combination with language constructs at different levels of granularity (word, phrase, sentence) to allow for flexibility in lexical and syntactic choice. We have explored how this new approach can be utilized for the application of indicative summary generation, which can summarize a book with a few sentences. We have detailed what types of semantic information (predicates) are present in indicative summaries and how a NLG architecture can utilize these resources to gen-

erate new summaries of unseen material.

We feel that this approach can be used to generate texts in other domains where the target texts exhibit strong regularity in content and its ordering, and that this approach can be paired with other techniques (such as heuristic-based sentence extraction) to apply to a wide range of texts.

References

Srinivas Bangalore and Owen Rambow. 2000. Exploiting a probabilistic hierarchical model for generation. In *Proc. of the 18th Intl. Conf. on Computational Linguistics (COLING 2000), Saarbrucken, Germany*.

Regina Barzilay and Kathleen R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *Proc. of ACL/EACL 01*.

Regina Barzilay, Noemie Elhadad, and Kathleen McKeown. 2001. Sentence ordering in multidocument summarization. In *Proc. of Human Language Technology '01*, San Diego, CA, USA.

Douglas Biber. 1989. A typology of English texts. *Linguistics*, 27:3–43.

William W. Cohen. 1995. Fast effective rule induction. In *Proc. 12th Intl. Conf. on Machine Learning*, pages 115–123. Morgan Kaufmann.

Michael John Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proc. of the 34th ACL*, Santa Cruz.

Edward T. Cremmins. 1982. *Art of Abstracting*. ISI Press.

Pablo A. Duboue and Kathleen R. McKeown. 2001. Empirically estimating order constraints for content planning in generation. In *Proc. of the ACL-EACL 2001*, Toulouse, France.

Min-Yen Kan and Judith L. Klavans. 2002. Using librarian techniques in automatic text summarization for information retrieval. In *Proc. of Joint Conf. on Digital Libraries*, Portland, Oregon, USA, July.

Min-Yen Kan, Kathleen R. McKeown, and Judith L. Klavans. 2001a. Applying natural language generation to indicative summarization. In *Proc. of 8th European Workshop on Natural Language Generation*, Toulouse, France.

Min-Yen Kan, Kathy McKeown, and Judith Klavans. 2001b. Domain-specific informative and indicative summarization for information retrieval. In *Proc. of the Document Understanding Conference (DUC)*, pages 19–26, New Orleans, USA.

Jussi Karlgren and Douglass Cutting. 1994. Recognizing text genres with simple metrics using discriminant analysis. In *Proc. of COLING '94*, Kyoto, Japan.

Brett Kessler, Geoffrey Nunberg, and Hinrich Schütze. 1997. Automatic detection of text genre. In *Proceedings of the 35th Association of Computational Linguistics (ACL '97)*, pages 32–38, Madrid, Spain.

Irene Langkilde. 2000. Forest-based statistical sentence generation. In *6th Applied Natural Language Processing Conf. (ANLP'2000)*, pages 170–177, Seattle, Washington, USA.

Library of Congress. 2000. Marc 21 format for classification data : including guidelines for content designation. Washington, D.C., USA. ISN 0660179903.

Elizabeth Liddy. 1991. The discourse-level structure of empirical abstracts: An exploratory study. *Information Processing and Management*, 27(1):55–81.

Inderjeet Mani and Mark Maybury, editors. 1999. *Advances in Automatic Text Summarization*. MIT Press.

Alice Oh and A Rudnicky. 2000. Stochastic language generation for spoken dialogue systems. In *Proc. of the ANLP/NAACL 2000 Wrkshp. on Conversational Systems*, pages 27–32, Seattle, Washington, USA, May.

Dragomir R. Radev. 1998. Learning correlations between linguistic indicators and semantic constraints: Reuse of context-dependent descriptions of entities. In *Proc. of COLING/ACL 98*, Montreal, Canada.

D. V. Rama and Padmini Srinivasan. 1993. An investigaton of content representation using text grammars. *ACM Transactions on Information Systems*, 11(1):51–75, January.

Adwait Ratnaparkhi. 2000. Trainable methods for surface natural language generation. In *Proc. of the 6th Applied Natural Language Processing Conf. (ANLP-NAACL 2000)*, pages 194–201, Seattle, Washington, USA.

James Shaw and Vasileios Hatzivassiloglou. 1999. Ordering among premodifiers. In *Proc. of the 37th Association for Computational Linguistics*, pages 135–143, College Park, Maryland, USA, June.

Sebastian Varges and Chris Mellish. 2001. Instance-based natural language generation. In *Proc. of NAACL 01*.