Knowledge-Based Report Generation: A Technique for Automatically Generating Natural Language Reports from Databases

Karen Kukich

Information Science Department
University of Pittsburgh
Pittsburgh,PA
and
Bell Telephone Laboratories
Murray Hill, New Jersey

Abstract. Knowledge-Based Report Generation is a technique for automatically generating natural language summaries from databases. It is so named because it applies the tools of knowledge-based expert systems design to the problem of text generation. The technique is currently being applied to the design of an automatic natural language stock report generator. Examples drawn from the implementation of the stock report generator are used to describe the components of a knowledge-based report generator.

Keywords. Natural Language Processing, Text Generation, Knowledge-Based Expert Systems, Databases

Databases and Natural Language Summaries

Among the growing number of machine-readable databases available for computer processing are periodic numeric databases. Examples of such databases include the Dow Jones stock quotes database, which contains half hourly quotes of over 1200 stocks on the New York Stock Exchange, the U. S. Weather Service meteorological database, which contains hourly weather statistics for over fifty weather stations throughout the country, a variety of statistics databases maintained by U. S. agencies such as the Department of Commerce, the Bureau of Labor and Unemployment, and the Department of Energy, and thousands of corporate databases containing inventory data, sales data, and equipment and facilities tracking data, maintained by large and small businesses.

The data in each of these machine-readable databases is processed by computer for a variety of applications, such as sorting by category, compiling totals and other statistics, and charting figures to indicate trends. The same data is also processed manually for at least one application, that of composing natural language reports that summarize and highlight points of interest in the data. Natural language reports are generated manually because computers do not yet have the ability to compose fluent English text. Whether a computer can be programmed to compose a natural language summary report from a database is the subject of this research.

The technique for designing a computer program to generate fluent natural language reports from databases is referred to as Knowledge-Based Report Generation because it is based on the principle that in order for a system to generate intelligent fluent text, it must incorporate a variety of types of knowledge, including domain specific semantic, linguistic, and rhetoric knowledge. The first application of the technique, a system for automatically generating natural language stock reports from daily stock quotes, is partially implemented, and will be used to illustrate the principles of the technique in the following discussion.

Natural Language Report Generation Problems

Given the task of composing an accurate, interesting and fluent summary of the data in a database, a person, or a system, must solve two problems: he must determine what to say and he must decide how to say it. Neither problem is trivial. A virtual infinity of facts can be inferred from the numeric data, but not all of them are interesting, nor would a random sample of them constitute an informative summary. Furthermore, the recitation of a series of simple facts would not satisfy the principles of rhetoric which govern fluent, mature text generation.

In determining what to say, for example, it will not do for a stock report generator to say simply:

> Dan River was up 3.25 to 26.
> DataPoint was down .5 to 13.
> DataGeneral was down 1.75 to 12.25.
> .
> .
> .

Instead, a summary should consist of statements such as:

> Even brisk trading in IBM and AT&T stock was unable to sumulate the Dow Jones average of 30 industrials which fell 3.82 points to 801.57.

Generating statements such as the one above requires some specific knowledge about the domain of discourse, in this case, the world of the stock market. In general, a summary writer must have the knowledge needed to filter out trivial facts and to recognize interesting points, points of comparisons, and trends. In particular, the writer of the statement above had to know that both IBM and AT&T stocks figure in the calculation of the Dow Jones average of 30 industrials.

Deciding how to express messages requires linguistic knowledge, including syntactic, grammatical, and rhetoric skills. The same messages might be expressed awkwardly in five monotonous sentences:

> The stock market closed lower. Many issues were down. Energy stocks were most active. Energy stocks were down. Trading was heavy.

or gracefully in one flowing sentence:

> Energy stocks bore the brunt of the selling pressure yesterday as the stock market suffered a broad setback in heavy trading.

Some of the linguistic knowledge required to generate fluent text is easily specified. For example, syntactic and grammatical rules governing such things as number agreement between subject and verb and pronoun case usage can be stated as formal principles. But the knowledge of rhetoric that is drawn upon to generate prose

But the knowledge of rhetoric that is drawn upon to generate prose that is clear, economical, varied, effective, and coherent is often difficult to make explicit. Teachers of rhetoric still rely on heuristics and examples to impart this form of linguistic knowledge to students.

Related Research: Text Generation and Knowledge-Based Expert Systems

Knowledge-based report generation lies at the intersection of two areas of research: text generation research and knowledge-based expert systems research. Text generation research is a relatively new area of interest within the field of natural language processing by computer. Knowledge-based expert systems research has recently achieved a good deal of success and fame.

Language generation has been divided into two major components by Thompson: a "tactical component", and a "strategic component".[1] The strategic component takes care of deciding what to say and how to say it; the tactical component takes care of the details of grammar and syntax, such as matching the number of the verb to the number of the subject of a sentence, and deciding when to use a pronoun. Early research in language generation, such as the work of Goldman[2] addressed the tactical problems. More recent research, such as the work of McKeown,[3] Conklin and McDonald,[4] and Mann and Moore,[5] has begun to address the strategic problems.

Goldman wrote a program to serve as the generation component of the language understanding system called MARGIE, which was designed by Schank and the Artificial Intelligence Project at Stanford.[6] MARGIE was a language understanding system which could analyze stories, answer questions based on them, and paraphrase sentences from them. In MARGIE, linguistic information was mapped into a conceptual representation scheme consisting of a relatively small set of semantic primitives. The conceptual representation scheme provided input to Goldman's program BABEL, which performed the task of selecting words to express the semantic primitives and organizing them into sentences. BABEL solved many of the tactical problems of language generation.

McKeown is one of the first language generation researchers to begin to address strategic processing problems. She has formulated a number of principles for use in the process of generating relevant answers to questions about database structure. These include strategies such as compare and contrast, top-down description, bottom-up description, definition, analogy, and illustration through example.

Conklin and McDonald are implementing a technique for generating verbal descriptions of scenes in photographs. Their technique for strategically planning the content and organization of text is based on the principle of visual salience. Objects in a photograph are manually ranked according to their visual salience, and an automatic planner formulates message descriptions based on the order of salience of objects and their relationships to each other.

The goal of the Knowledge Delivery System, or KDS, of Mann and Moore, is to deliver knowledge extracted from large knowledge bases packaged in the form of multi-sentence output. Mann and Moore consider the major strategic problem of the system to be deciding what to say and what not to say. Thus, after using a fragment-and-compose process to break the knowledge in the database into manageable units and reorganize it into meaningful messages, They apply a knowledge filter to prevent the system from expressing everything.

All of the above research projects address important and difficult aspects of the text generation problem. The major characteristic that distinguishes the knowledge-based report generation technique from other text generation projects is its goal of starting directly from the numeric data for its input to the system. All other text generation systems are designed to start with information that has been encoded into some knowledge representation formalism such as a semantic network.

Because it is clear that a natural language report generator must make use of much semantic and linguistic knowledge for the domain of its discourse, it is only natural to turn to research in knowledge-based expert systems to look for techniques for knowledge representation. In fact, knowledge-based expert systems do provide elegant tools for knowledge representation in the form of production system languages. In production system languages, domain-specific knowledge is represented in small knowledge packets called production rules. A production rule is really a miniature program in the form of a pattern-action element. It tells the computer, "if you see a certain pattern in the data, take the following action". An expert system is a set of anywhere from tens to hundreds of these pattern-action rules, or knowledge packets, that attempt to represent an expert's knowledge in a particular domain. For example, expert systems have been designed to diagnose diseases,[7] to predict locations of mineral deposits,[8] to configure computer hardware components,[9] and to configure computer processors on VLSI chips.[10] For an overview of knowledge-based expert systems, see Kowalski [11] or Duda and Gaschnig.[12]

Theory Underlying Knowledge-Based Report Generation

The technique of knowledge-based report generation is based on the premise that a variety of types of knowledge are brought to bear during the process of generating a natural language summary. That knowledge includes both semantic and linguistic knowledge for the particular domain of discourse in which text is being generated. The notion that a domain of discourse has its own sublanguage is the first tenet of knowledge-based report generation. The existence of macro-level knowledge structures for both semantic and linguistic knowledge is a second tenet.

A sublanguage, as introduced and defined by the linguist Zellig Harris,[13] is a proper subset of the sentences of a language that is closed under some or all of the operations defined in the language. The most extensive research into the nature of sublanguages has been done by Kittredge and Lehrberger while working in the area of machine translation.[14] By analyzing representative samples of texts from specialized fields, such as weather reports, stock reports, aviation hydraulics manuals, pharmacology reports and others, Kittredge, Lehrberger, and colleagues have identified distinguishing grammatical traits, such as frequency of occurrence of relative clauses, verb tense dominance, use of synonyms and hyponyms, etc., which characterize each sublanguage. The extent and usefulness of sublanguage description is clarified by Kittredge:

> Although no sublanguage has been described in all its details, a relatively complete description for the sublanguage of weather reports has led to the design of the first translation system whose output does not have to be revised. (p. 1)[14]

Not only does a sublanguage circumscribe the linguistic boundaries of a domain of discourse by establishing the lexicon and delineating the permissible grammatical forms, but it also suggests the semantic boundaries of the domain of discourse. Again, Kittredge points out:

> The lexical classes and the hierarchical relations between the classes usually reflect the accepted taxonomy which the specialized field of knowledge imposes on the objects of its limited domain of discourse. And the combinations of lexical classes which are permissible in the sentences of the specialized texts reflect conceivable relations between these objects. (p. 8)[14]

Sublanguage knowledge of both the semantics and the linguistics of a particular domain of discourse is precisely the knowledge that must be incorporated in a knowledge-based report generator. Furthermore, it is sublanguage knowledge that provides the constraints that make natural language text generation computationally feasible. In short, sublanguage knowledge is a necessary and sufficient condition for natural language text generation.

The second tenet of knowledge-based report generation, the existence of macro-level knowledge structures for both semantic and linguistic knowledge, also contributes to the computational manageability of a report generator. By macro-level knowledge structures is meant higher-order units of both semantic and linguistic knowledge, such as phrases rather than words, semantic messages rather than semantic primitives, and a clause-combining rather than clause-generating grammar.

Joseph Becker first introduced the concept of a phrasal lexicon in 1975:

> I suggest that utterances are formed by the repetition, modification, and concatenation of previously-known phrases consisting of more than one word. I suspect that we speak mostly by stitching together swatches of text that we have heard before; productive processes have the secondary role of adapting old phrases to the new situation. (p. 70)[15]

He goes on to cite examples of phrasal lexical items from the text of his own article. They include: "this is not to say that", "to sweep under the rug", "as (something) should make apparent", (verb) the un(verb)able, and others.

Notice some things about the entries in the phrasal lexicon. First, they are not always literals. Some entries contain variables, such as "something" in "as (something) should make apparent". Second, definitions of phrasal lexical items correspond to higher-order semantic units, whole messages as opposed to semantic primitives. Third, generating text from whole phrases requires higher-order grammatical rules, such as clause-combining and clause-transforming rules, as opposed to clause-construction rules. These three macro-level constructs, a phrasal lexicon, a knowledge base of conceptual messages, and a clausal grammar, make it possible to implement a computer system that incorporates the semantic and linguistic knowledge of a sublanguage within a computationally manageable framework. They provide the underlying motivation for knowledge-based report generation.

The technique of knowledge-based report generation should not be construed as an attempt to model human natural language production. In fact, some of the design constraints are probably psychologically invalid. For example, as will be discussed shortly, a knowledge-based report generator consists of five independent sequential modules. Each module performs a separate task, such as inferring semantic messages, organizing messages into paragraphs, and generating linguistic strings to express those messages. The modules operate sequentially and there is no feedback or backup processing. The report generator was designed with independent sequential modules for the sake of computational manageability, and it probably does not reflect the way people generate and organize text. There is some evidence that people generate and organize verbal speech in such a sequential, right-branching fashion, however. Verbal speech occasionally contains sentences that are aborted or syntactically redirected in mid-stream because the speaker either finds himself at a dead-end or conceives of a better way to complete a thought. This phenomenon is called anacoluthia.

Despite the fact that a knowledge-based text generator incorporates some design constraints that negate its psychological validity as a model of language generation, the technique may be viewed as a first step towards a general theory of language generation. In particular, the two fundamental tenets of the technique, the need for domain-specific sublanguage knowledge, and the use of macro-level structures and processes, probably *are* psychologically valid, and they must be accounted for by a general theory of language generation.

I propose that a general theory of language processing must view language generation as a multi-level process. The metaphor of shifting gears while driving a car provides a useful analogy for understanding multi-level language processing. Just as driving in third gear makes the most efficient use of an automobile's resources, so also does generating language in third gear make most efficient use of human information processing resources. That is, matching whole messages to whole phrases and applying a clause-combining grammar is cognitively economical. But when only a near match for a message can be found in a speaker's phrasal dictionary, the speaker must downshift into second gear, and either perform some additional processing on the phrase to transform it into the desired form to match the message, or perform some processing on the message to transform it into one that matches the phrase. And if not even a near match for a message can be found, the speaker must downshift into first gear and either construct a phrase from elementary lexical items, including words, prefixes, and suffixes, or reconstruct the message.

As currently configured, a knowledge-based text generator operates only in third gear. Operating exclusively in third gear poses both advantages and disadvantages. The main advantage is a gain in computational economy and text quality. Because the units of processing are linguistically mature whole phrases, the report generation system can produce fluent text without having the detailed knowledge needed to construct mature phrases from their elementary components. The intrinsic semantic and linguistic constraints of the sublanguage make this possible. The main disadvantage of macro-level language generation is that the system lacks much of the flexibility of human language generation capabilities. A macro-level generator can only generate predefined messages and phrases; it cannot combine messages attributes or sub-phrasal linguistic units (such as words) in novel ways.

But a knowledge-based report generator need not be confined to operating in third gear forever. Because a knowledge-based report generator is implemented in an easily modifiable production system language, it may be viewed as a starting tool for modeling and extending a a theory of multi-leveled language generation. By experimenting with additional knowledge, a knowledge-based report generator could gradually be extended to shift into lower gears, and to exhibit greater interaction between semantic and linguistic components.

Components of a Knowledge-Based Report Generator

A knowledge-based report-generator is a computer software system that makes use of expert-system techniques for representing the semantic and linguistic knowledge needed to generate fluent natural language reports from numeric databases. By expert-system techniques is meant that the knowledge of a specific domain is represented in production rules, or knowledge packets, which recognize patterns and fire appropriate actions. A different generator must be implemented for each different report domain, such as the domains of stock reports, weather reports, or corporate reports. This is necessary because the semantic and linguistic knowledge of each different report domain forms its own sublanguage, which includes conceptual messages, a lexicon, and a grammar. The control mechanism of a knowledge-based report generator, however, is domain independent, and accounts for about twenty percent of the system.

A knowledge-based report generator consists of five independent sequential modules: a Fact Generator, a Message Generator, a Discourse Organizer, a Predicate Text Generator, and a Polished Text Generator. (see Figure 1) Each module is a filter, ie., a program that accepts some input, performs some process on it, and produces some output, all without any interaction with a user or another program. The input to the first module is data from the numeric database; the input to each subsequent module is the output of the preceding module; the output of the final module is the finished report. There is no human intervention from the time the numeric data is fed into the system until the time the finished report is turned out. In the following description of the function of each of the five modules, examples are drawn from the Stock Report Generator which is partially implemented.

The first module, the Fact Generator, performs the simple task of extracting data from the numeric database and computing relevant statistics. For example, closing averages are extracted for all stocks, and directions and degrees of change are computed, as are averages for groups of stocks such as oils, retails, auto stocks, etc. The output of the first module is a set of facts which represent the pertinent statistics for the report period.

Knowledge-Based Report Generator Components

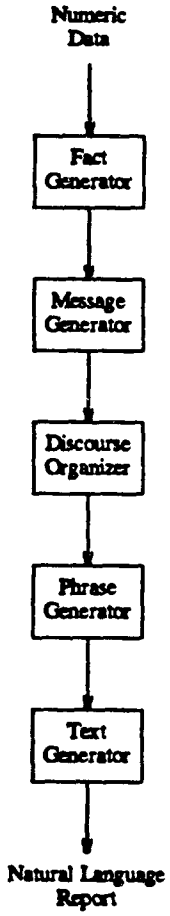

Figure 1

The goal of the second module, the Message Generator, is to instantiate potential messages by drawing semantic inferences from the facts. Potential messages are represented by conceptual message templates. Inferences are drawn by production rules that recognize patterns in the facts, and messages are instantiated when recognized patterns fire actions that assign values to the attributes of message templates. The output of the Message Generator is a set of semantic messages that reflect the significant events for the report period. For example, a message template indicating the closing status of the market, its direction, degree, and scope of change, will always be instantiated. One or more message templates indicating interesting events during the day, such as sudden surges or plunges, or record high or low prices, may be instantiated.

Module three is the Discourse Organizer, whose task is to determine the order in which messages are to be expressed. This task calls for discourse structure knowledge, which is again represented by production rules. Discourse structure rules assign both topic importance values and message importance values to messages. Then they compute priority values for messages as a function of topic importance and message importance. Roughly, all messages of the same topic will be grouped into a single paragraph, and messages will be ordered within paragraphs according to their importance values. There will be exceptions in which an unusual event, such as the Dow Jones average hitting a record high, will be given top billing outside its topic paragraph. The output of the Discourse Organizer is a set of prioritized semantic messages.

Prioritized conceptual messages form the input to phase four, the Predicate Text Generator. This module performs the most complicated processing. Its task is to map conceptual messages into predicate text which has the structure and content of fluent text with only the choice of lexical subject, and consequently verb number ending, left unresolved. It accomplishes this by selecting an appropriate phrase from the phrasal dictionary, deciding on an appropriate syntactic form for the phrase, such as sentence, relative clause, or prepositional phrase, and combining the phrase with foregoing phrases to form mature, (i.e., complex) grammatical sentences. If more than one phrase matches the conceptual message, the Predicate Text Generator makes use of rhetoric rules governing such things as sentence length to select from among them. If more than one syntactic form is available, it makes use of rhetoric knowledge governing such things as use of varied syntax to select an acceptable syntactic form. The Predicate Text Generator incorporates a clause-combining grammar to transform phrases into mature sentences. All of the knowledge of the Predicate Text Generator is embodied in production rules.

Finally, the fifth module, the Polished Text Generator, takes the predicate phrases generated by the fourth module, and converts them to polished text by performing such tasks as choosing appropriate lexical subjects, such as nouns, pronouns, or nothing in the case of ellided subjects, and selecting the singular or plural version of the verb as appropriate. The output of module five is the fluent natural language report.

Implementing a Knowledge-Based Report Generator

Because the system must be instilled with a great deal of domain-specific semantic and linguistic knowledge, the task of implementing of a knowledge-based report generator is neither quick nor automatic. Using the shell of a knowledge-based report generator from another domain will reduce the amount of implementation effort required, but the steps remain the same. They include the following:

1) Analyze a sample of manually generated reports to identify the phrasal units and syntactic forms of the sublanguage

2) Analyze the same representative sample of manually generated reports to identify the message classes and semantic attributes of the sublanguage

3) for a subset of semantic messages, begin building both subject and predicate phrasal dictionaries incorporating the semantic messages and syntactic forms of the sublanguage

4) for the same subset of semantic messages, create the knowledge packets (production rules) for generating relevant facts

5) for the same subset of semantic messages, create the knowledge packets (production rules) for instantiating messages

6) for the same subset of semantic messages, modify the knowledge packets (production rules) for organizing discourse

7) for the same subset of semantic messages, modify the knowledge packets (production rules) for generating predicate text

8) for the same subset of semantic messages, modify the knowledge packets (production rules) for generating polished text

9) reiterate steps 3 through 9 for all semantic messages

Expert Status

Despite the fact that a knowledge-based report generator is implemented in expert system software, it does not necessarily follow that it is an expert system. A knowledge-based report generator incorporates only as much knowledge as its implementor instills in it. For the purpose of generating summary and highlight reports, only general knowledge, as opposed to expert knowledge, is required of the system. So for example, a stock report generator could not "take a position" on a stock, or advise a client about when to sell short.

However, because the knowledge in a knowledge-based report generator is easily modified and augmented, there is nothing to preclude the system from being gradually upgraded into an expert. In the case of the stock report generator, for example, the Fact Generator module might be upgraded to perform cluster analyses, to chart cycles, and to detect trends. If the remaining modules were to be imbued with the appropriate semantic messages and technical jargon, the system might be converted into a technical analyst, i.e., a particular type of stock market expert.

Alternatively, the stock report generator could be modified to produce reports tailored to individual or corporate interests by closely watching particular stocks or groups of stocks. Or "timely" reports might be produced by having the system prompt the user at start up time for names of companies in the news.

1. H. Thompson, "Strategy and Tactics: A Model for Language Production," in *Papers from the 13th Regional Meeting, Chicago Linguistic Society* (1977).
2. Neil M. Goldman, "Sentence Paraphrasing from a Conceptual Base," *CACM* **18**(2) (February 1975).
3. Kathleen R. McKeown, *Generating Relevant Explanations: Natural Language Responses to Questions about Databases,* Proc. 1980 AAAI Conf. (August 1980).
4. E. Jeff Conklin and David D. McDonald, "Salience as a Simplifying Metaphor for Natural Language Generation," pp. 75-78 in *Proceedings of the National Conference on Artificial Intelligence*, Pittsburgh, PA (August 1982).
5. James A. Moore and William C. Mann, "A Snapshot of KDS: A Knowledge Delivery System," in *Proceedings of the 17th Annual Meeting of the Association for Computational linguistics*, La Jolla, California (11-12 August 1979).
6. Roger C. Schank, *Conceptual Information Processing* , North Holland, Amsterdam, The Netherlands (1975).
7. Janice S. Aikins, "Representation of Control Knowledge in Expert Systems," pp. 121-123 in *Proceedings of the First Annual National Conference on Artificial Intelligence*, The American Association for Artificial Intelligence, Stanford University (August, 1980).
8. A. N. Campbell, V. F. Hollister, R. O. Duda, and P. E. Hart, "Recognition of a Hidden Mineral Deposit by an Artificial Intelligence Program," *Science* **217**(4563), pp. 927-929 (3 September 1982).
9. John J. McDermott, "R1: A Rule-Based Configurer of Computer Systems," *Artificial Intelligence* **19**, pp. 39-88 (1982).
10. T. J. Kowalski and D. E. Thomas, *The VLSI Design Automation Assistant: First Steps,* Carnegie-Mellon University Electrical Engineering Department (1982).
11. T. J. Kowalski, "Knowledge-Based Expert Systems: An Introduction," 81-11229-19, Bell Laboratories Technical Memorandum (October 15, 1981).
12. Richard O. Duda and John G. Gaschnig, "Knowledge-Based Expert Systems Come of Age," *Byte*, pp. 238-281 (September 1981).
13. Zellig Harris, *Mathematical Structures of Language*, Interscience Publishers, John Wiley & Sons, New York (1968).
14. Richard Kittredge and John Lehrberger, *Sublanguages: Studies of Language in Restricted Semantic Domains*, Walter DeGruyter, New York (in press).
15. Joseph Becker, "The Phrasal Lexicon," pp. 70-73 in *Theoretical Issues in Natural Language Processing*, ed. B. I. Nash-Webber, Cambridge, Massachusetts (10-13 June 1975).