THE EXPERIENCE OF DEVELOPING A LARGE-SCALE NATURAL LANGUAGE TEXT PROCESSING SYSTEM: CRITIQUE

Stephen D. Richardson and Lisa C. Braden-Harder

IBM Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, New York 10598

Abstract

This paper describes our experience in developing the CRITIQUE system. It describes three application areas in which the system is being used and discusses some characteristics of CRITIQUE which we believe are applicable to large-scale natural language systems in general: performance, robustness, flexibility, presentation, and accuracy.

Introduction

CRITIQUE is a large-scale natural language text processing system that identifies grammar and style errors in English text. This advanced prototype, which is currently being developed at IBM Research, is based on a broad-coverage natural language parser (Richardson, 1985). The parser provides a unique approximate syntactic parse for a large percentage of English text and diagnoses over 100 grammar and style errors.

Earlier writing-aid systems, such as Writer's Workbench (Macdonald, et al, 1982), contain functions which identify parts-of-speech of words, perform string-level phrase identification, and generate readability statistics. Similar functions are apparent in many systems now commercially available, some of which are described in an issue of *The Seybold Report* (1984). To our knowledge, however, no other system uses a parser that produces complete structural analyses for sentences.

CRITIQUE is an extension of the EPISTLE project which began in 1980 (Heidorn, et al, 1982). The parser and grammar are implemented in PLNLP (the Programming Language for Natural Language Processing), developed by George Heidorn. PEG (the PLNLP English Grammar) has been written by Karen Jensen, and the style rules were written by Yael Ravin. Today, CRITIQUE is being tested in a variety of applications ranging from office correspondence and technical documentation to student essays. Also, PLNLP and PEG have been incorporated into several other research applications, such as machine translation systems.

At the 1986 ACL meeting, Gary Hendrix described his experience in developing a natural language interface for real users (Hendrix, 1986). In contrast with ***user interface*** systems, we consider CRITIQUE to be a ***text processing*** system. The latter may be distinguished from the former by its broad coverage of texts that were prepared independently to communicate ideas and not strictly to interact with a computer system. Until now, the experience of developing a large-scale natural language text processing system has not been discussed in the literature.

This paper first describes the overall processing in the CRITIQUE system. Then it describes three application areas in which the system is being used. The remaining sections discuss some characteristics of CRITIQUE which we believe are applicable to large-scale natural language systems in general: performance, robustness, flexibility, presentation, and accuracy. The discussion draws on our experience in all three application areas.

Processing in CRITIQUE

CRITIQUE processes text in six steps. The first step determines sentence, heading, and paragraph boundaries. In the next step, lexical processing identifies unrecognized words and awkward phrases. The on-line dictionary which is used includes more than 100,000 entries and provides information used in syntactic processing as well. After lexical analysis, text is passed to the parser, which produces a parse tree, and in so doing checks for grammar errors. Then stylistic analysis diagnoses potential style prob-

lems. CRITIQUE also generates statistical information about documents based on the lexical and syntactic analyses. The final step involves error summarization and display.

CRITIQUE has an interactive processing mode that is fully integrated with a text editor, allowing users to update the text as needed. As the text is modified, new sentences are reanalyzed to ensure that no new errors have been introduced. The system provides three levels of on-line help: the first level identifies the error, the second provides a brief explanation, and the third provides a complete tutorial. Figure 1 is an illustration of the second level of help. The user can also specify style preferences in an individual profile. Possible errors are filtered through the profile to determine whether or not they should be displayed. Hard-copy output is also available.

I am writing to recommend Susan Hayes, who's application you recently received.

Confusion of "who's" and "whose"
whose
The word "who's" (which means "who is") and the word "whose" (which is possessive) cannot be interchanged.

Figure 1. Second level of Help includes name of error, suggested correction, and a brief explanation

Application Areas for CRITIQUE

During the development of CRITIQUE, we have directed our efforts towards three major application areas: office environments, publications organizations, and educational institutions. Each area has its own particular needs and requirements.

In the office environment, professionals require quick, succinct feedback on their memos and other documents. They are less interested in maintaining a particular style, but want insurance against obvious grammatical and spelling mistakes. Our parsing grammar was originally developed using a data base of office correspondence. There has also been an abundance of feedback at IBM Research, where the system has been made available to hundreds of users. These users submitted over 3,000 pages of text to CRITIQUE in 1987.

Publications organizations usually have strict requirements for style and consistency which exist in the form of tedious style guides. The professional writers in such organizations also want succinct feedback, but are usually willing to wait longer to receive it, since their documents are typically longer and more involved. An IBM technical writing group and the US government have been our source of experience and feedback in this area.

Use by educational institutions has proven to be the most challenging of the three areas. There is a wide range of ill-formed text to deal with, originating from classes in composition, business writing, technical writing, and ESL (English as a Second Language). The professors in these various areas also sometimes have differing opinions on grammar and style. Although there may not be such a great need for quick processing time (except by those students who procrastinate), processing cost must be minimized to fit most university budgets. We currently are doing joint studies with three universities to help test and refine CRITIQUE.

Performance

Broad-coverage natural language processing is computationally expensive. To do it in real time is even more so. Whereas large offices and publications organizations may be able to afford extensive computing power, such is not the case in many of the environments where a system such as CRITIQUE would be most useful.

Although CRITIQUE has been developed in a large IBM-mainframe environment, several significant steps have been taken to improve its performance with a view toward running on much smaller machines. In addition, a version of the PLNLP parser on which CRITIQUE is based was successfully ported to an IBM PC in the summer of 1986. Work is continuing on other versions which would run the complete PLNLP English Grammar (PEG) on intelligent workstations such as the IBM RT PC and PS/2.

We have used two complimentary approaches to achieve satisfactory performance. One is to distribute the parts of the system which can run

in parallel over multiple processors (where available), and the other is to optimize the performance of the programs themselves.

To distribute the processing involved, we have used "parsing server" programs which may operate either on the same physical computer, or on several computers connected by a network. When CRITIQUE is invoked by a user, each sentence in the user's document is sent as a separate task to a "manager server" which then distributes such tasks to as many parsing servers as are available. After analysis, information about a sentence is returned via the manager server to the user's editing environment. With this scheme, multiple users can access multiple parsing servers that may reside on different computers linked by a network. In this way, several of a user's sentences may be processed in parallel and asynchronously with respect to other tasks (such as word processing) that the user may be doing.

Although this distributed processing system is currently implemented on a network of mainframes, the transition to a workstation-based network like those found in small businesses and university environments will not be difficult. The distributed architecture is also well suited to exploit the power of parallel processor machines currently under development. The granularity of the processing involved, which is now at the sentence level, may also be made smaller or larger, depending on resulting efficiency and the possible need to consider larger segments of text for a more complete analysis.

The parsing servers referred to above consist of the PLNLP parsing engine, the PLNLP English Grammar, and a large set of style rules. PLNLP supports the writing of procedures as well as rules. Consequently, the parsing engine itself is written as a set of PLNLP procedures. In addition to the run-time environment, the translator for the PLNLP language is also written using PLNLP. When an entire programming language system such as this is written in itself, a high degree of portability and language-specific optimization may be achieved, further enhancing overall system performance.

The PLNLP translator currently turns PLNLP rules and procedures into LISP or PL.8 (a highly-optimized PL/I variant) code which is then compiled and executed. Work has also been done using C as a base (for the PC version mentioned earlier), and this work will be extended for portability across computers. Direct compilation into machine code is also being considered.

In our experience with various programming languages and environments, we have found it desirable to maintain two versions of the system, which share the same PLNLP source code. One is geared toward grammar and style rule development, being somewhat slower, but very flexible, and containing a set of specially designed tools and development aids. This version of the system now runs in LISP. The other version, running in PL.8, is optimized for fast execution and is about ten times faster than the development version. CRITIQUE uses the PL.8 version, which can analyze a sentence of about 15-20 words in one CPU second on an IBM 3081 computer. This translates into a few seconds of elapsed time under an average load.

Even as computers become more powerful, there will continue to be a corresponding increase in the complexity and amount of computation involved in natural language processing. Through use of a highly-optimized production run-time environment, PLNLP is able to achieve the required performance without sacrificing flexibility during development.

One last performance issue should be mentioned: the need for a well-integrated dictionary system. As previously stated, CRITIQUE's dictionary is able to recognize well over 100,000 words, providing both morphological and syntactic information about those words. The trade-offs between keeping the dictionary on disk or in memory are more significant in a very large-scale system. Disk I/O's, including "hidden" paging I/O's when the dictionary is in virtual memory, must be carefully considered and minimized. It has been our experience that expensive dynamic morphological processing should also be kept to a minimum, although this may not be possible for other languages.

Robustness

Any computer system should be robust. This is especially true of natural language systems, and, in particular, those which specialize in handling ill-formed input. Robustness should be considered at every level of processing, both for

the system in general and for the particulars of dealing with natural language inputs.

At the system level, the distributed architecture which is used by CRITIQUE for performance reasons requires robust task management mechanisms. The manager server carefully tracks the progress of each task (sentence) and the availability of parser servers on the network. If a parser loses its network connection, exceeds a predetermined time limit, or otherwise fails while processing a task, that task is sent out again to another parser. If a parser fails while processing a task, it automatically restarts itself. Statistics concerning usage and task flow, as well as comments recorded by users about the usefulness or accuracy of critique information, are maintained by the manager server and automatically distributed to system developers each day.

At the natural language level, robustness first comes into play in handling the various formats of text inputted to the system. Text which has been "manually" formatted (using an editor, "WYSIWYG" style), as well as text with imbedded formatting commands (IBM's SCRIPT and GML commands are currently supported) is scanned by CRITIQUE to identify "parsable" segments. This process excludes tables, figures, headings, addresses, etc., and is table driven to accommodate the varying requirements of users in the different application areas. Publications organizations, for example, typically have special additional sets of formatting commands that must be supported.

During parsing, words which are not in the dictionary are assigned default morphological and syntactic information so as to avoid a parsing failure. Most such words are generally assumed to be singular nouns, although there are some exceptions. This is usually adequate to obtain a reasonable parse, but can cause problems when it is a verb that is misspelled.

Parsing may take place in one pass or two, if necessary. The first pass applies the rules of the grammar with all of the constraints in force. If a parse is not obtained, then a second pass is made, applying the rules with selected constraints being relaxed. Certain lexical substitution rules for easily confused words (e.g., whose/who's, its/it's) are also activated during the second pass. If a parse is still not obtained after the second pass, whether because of an unanticipated error, an unrecognized word, or a possible weakness in the grammar, then the "parse fitting" procedure is invoked (Jensen, et al, 1984). This procedure relies on the fact that the parsing algorithm is bottom-up in nature, and therefore intermediate well-formed parse structures are produced for segments of the sentence. These structures may be "fitted" together to form a parse for the sentence if no other complete structure is found. Even when a fitted parse is obtained, grammar and style error detection is still active within the successfully parsed segments.

If multiple parses are obtained, the system selects one based on a parse metric which favors trees in which modifying words and phrases are attached to the closest qualifying constituent (Heidorn, 1982). If the number of parses obtained exceeds a certain threshold, CRITIQUE takes advantage of the situation and informs the user that the sentence is probably unclear. If the parser fails for some system reason, the user will receive a message that the segment of text in question was "too difficult to process."

No one can foresee all the errors that humans can make. It is for this reason that we have included these robust mechanisms, and that we continue to enhance the system to catch new errors as experience and feedback dictate.

Flexibility

By virtue of the significantly different needs of each application area listed earlier, flexibility has been a requirement throughout the development of CRITIQUE. For example, the publication organizations we have dealt with have required large additions of terminology, the handling of special input formats and formatting commands, and additional style critiques dictated by organizational style guidelines. Universities, being pedagogically oriented, have been very much concerned with the format and content of the critique information presented in the output. We have attempted to handle the need for this flexibility at the individual, installation, and application area levels.

The basic CRITIQUE system provides predetermined critiques, intuitively organized into groups, with default thresholds, if applicable, and general help and tutorial information. It handles

the formats of files by default according to certain file naming conventions. The vocabulary in the dictionary comes mainly from Webster's 7th Collegiate dictionary, and the grammar and style error rules have been developed according to several widely accepted sources. Every item of information produced by the system is controlled by a switch or threshold contained in a user profile. We have found that a good set of defaults in this profile is indispensable, since most users often do not bother to change them.

Individuals who use the system are free to change any of the settings in the profile according to their own tastes and needs. They may also add words to an addendum which is used solely for the purpose of checking spelling.

Knowledgeable users, or, more commonly, installation administrators, may change the default settings in the system profile or create several profiles for different purposes. Such would be the case for university classes of different types or various publications groups, each with its own particular style requirements. This level of customization also includes changing the grouping of critiques and the associated code (used by the system to flag the occurrence of an error in the output), message, help, and tutorial information, and making large additions of specialized terminology to the system dictionary. New classes of word- and phrase-level errors may be added to the dictionary as well.

Users at some of our test sites have requested the ability to add classes of style errors. This is not currently possible, because they would have to be able to write their own PLNLP rules. For now, further types of customization, for entire application areas, for example, are performed by the system developers, although there is continual re-evaluation of where to draw the line.

It is important to point out that the kinds of system customization described above have not, thus far, included tuning the grammar for special handling of the texts common in a particular application area. Every effort has been made to keep PEG as broad-coverage as possible. In fact, there has been a tendency during the development of CRITIQUE to move certain types of error detection, where possible, from the grammar to the style rule component. Since style rules are applied only after a parse has been obtained by the grammar rules, this lessens the possibility that testing for an error will interfere with grammar rule processing.

Presentation

Systems such as CRITIQUE are generally used to process texts which have been prepared for a human audience often using word processing software. Therefore it seems natural, perhaps even necessary, that these systems be tightly integrated with a word processing environment.

The CRITIQUE system architecture, which has been described previously from a distributed processing standpoint, may also be viewed as incorporating a word processing environment as a user interface, with a background natural language processor. There is nothing in the CRITIQUE system interface that requires that what the parser servers return be grammatical and stylistic information. The "descriptors" produced by the parsers are general in nature and could be used to send back any kind of information, possibly including a content characterization for information retrieval purposes or even a translation into another language. In this way, the system may be considered as a general purpose natural language processing environment.

With respect to the presentation of critique information in this integrated environment, the differing needs of the application areas have been evident once again. Several lessons in human factors have been learned and the results implemented.

In a prior version of CRITIQUE, problems were simply underlined on the screen, and users were required to point to a particular problem and request that a window be opened which contained a description of what was wrong. As a result of studying the usage statistics gathered by the manager server at IBM Research, we determined that users were not asking for the descriptions of errors. Instead, they seemed to rely on their intuitions, only making use of the fact that CRITIQUE had flagged a particular word or phrase. This led us to replace the underlining with a brief, highlighted code word or phrase which indicates what the problem is. In cases where CRITIQUE suggests a corrected form of a word, that form is now used as the error indi-

cator. This new format for displaying errors is shown in Figure 2.

Lets contemplate how a president is selected.
Let's
In many cases the best candidate in the eyes of
|MISSING COMMA
the public is the one who has the most exposure.
This is no way to chose a president, but
**choose*
unfortunately it is often true. The total package
of a candidates political ideas don't really make
**doesn't*
an impression on the public. His appearance
|FRAGMENT
and mannerisms and the amount of exposure
that make him successful.

Figure 2. Example of errors flagged by CRITIQUE

From this experience and other similar ones, we concluded that professionals using CRITIQUE in an office environment preferred a quick, interactive review of memos and documents. The amount of feedback on the screen at any one time should be maximized, and the number of keystrokes and overall review time thereby minimized.

Publications organizations have proved similar in many respects. However, due to the length and complexity of documents produced in such organizations, users may be more willing to wait for their output, and often make use of overnight batch runs.

One feature of CRITIQUE that has proved useful in this respect is called "interactive review." It is based on the fact that the system saves all of the information produced about a given file on disk at the end of a session or run. This information is then read the next time the same document is processed, thereby eliminating the need to reprocess sentences that have not changed. This means that it is possible for very large files to be run overnight, and then be reviewed interactively the following day, thereby lessening the impact on prime shift computer usage.

Publications groups, through their occasional use of sub-contractors that do not have access to on-line information, provided part of the motivation to optionally produce printed output which is almost identical to what is viewed on the screen. They also required the flexibility of easily integrating the information contained in an organizational style guide with the interactive tutorials for each critique.

The universities we are working with considered the abbreviated presentation of critique information we developed to be appropriate for their advanced students, but inadequate for others. They want the ability to lengthen explanations where desired and to group critiques by type, only presenting certain types in the output at any one time.

Our varied experiences in these application areas have resulted in highly flexible, table-driven presentation modes for both batch and interactive output. We continue to experiment and make changes based on feedback.

Accuracy

Accuracy is perhaps the most important aspect of a natural language system's overall performance. It may be evaluated from two perspectives: the actual "under-the-covers" natural language processing involved, and the user's perception. Given the state of the art, we may consider it a blessing that it is possible for the latter to be somewhat better than the former.

From a processing perspective in CRITIQUE, we reiterate that the PLNLP English Grammar produces parses which are approximate. Without recourse to semantics we cannot hope for much better. However, we are quite pleased with the coverage and accuracy that we have obtained, and find them to be adequate for the requirements of a system like CRITIQUE. The semantic ambiguities and inaccuracies which remain in the parses have not been a stumbling block to the usefulness of the system. This demonstrates that some degree of inaccuracy at the natural language processing level can be acceptable as long as it is not readily visible to the user. We do not pretend to be completely satisfied with this situation, however, and we are doing research in the area of "dictionary-based" semantic analysis. This will enable us to improve some of the attachments in the parse trees produced by PEG (Binot and Jensen, 1987).

Even being able to deal with a wide range of ill-formed input, it cannot be expected that a parser without a sophisticated semantic component can successfully parse "gobbledegook." In the goal to produce a useful and accurate analysis of text, there must also be an assumption included about the maximum degree of ill-formedness that can be handled. The sentence given below in Figure 3, which was taken from a real student essay, illustrates the kind of ill-formedness which challenges CRITIQUE to its limits. The system did point out the comma splice in this sentence, but nothing else.

"He starts to condemn Nora for her mistake and made as if she is like poison that can be contagious, Trovald was ready to take away the kids and kick Nora out as an outcast as how they did with Mr. Krogstad."

Figure 3. An ill-formed sentence from a student essay

In discussing the robustness of the parser, it was pointed out that error detection is still performed within the successfully processed segments of a fitted parse. Our testing to this point indicates that critiques produced in such situations are about as accurate as those produced in non-fitted parses. This is another case where the user's perception may differ from the underlying performance of the system.

In general, however, we have found users' perceptions and feedback to be most helpful. The facility that CRITIQUE provides for giving feedback allows users to classify advice provided by the system according to the categories ***correct***, ***useful***, ***missed***, and ***wrong***. These categories are self-explanatory except, perhaps, for the ***useful*** category. This refers to the case where a critique is not exactly correct; but, since the user's attention is drawn to a particular phrase or sentence, a real problem is noticed. We tend to include these kinds of critiques with those that are correct in evaluating the usefulness and accuracy of the system.

The most undesirable critiques are those in the ***wrong*** category, as they tend to destroy user confidence in the system and are not well tolerated in educational environments. We have found, however, that professionals seem much more forgiving of wrong critiques, as long as the time required to disregard them is minimal. This is similar to using spelling checkers, which wrongly highlight many proper names, acronyms, etc., but are still considered quite useful.

In order to analyze CRITIQUE's current accuracy in an educational environment, we recently processed a number of student essays provided by the computer-aided writing program at Colorado State University. We randomly selected 10 essays from each of four groups: freshman composition, business writing, ESL (English as a Second Language), and professional writing. The diagnoses made by CRITIQUE in these essays were reviewed and classified according to whether they were ***correct***, ***useful***, or ***wrong***. We did not consider errors that were ***missed***, but simply concentrated on the correctness of the critiques actually provided by the system. The reason for this orientation was our concern with the potentially damaging effect of wrong advice.

We adjusted the analysis in both directions, in a manner that we believe is fair. On the one hand, we did not count correct critiques of a trivial or mechanical nature, such as misspelled words, superficial punctuation checks, or readability scores. On the other hand, we also did not include a particular class of incorrect comma critiques, the handling of which we need to improve. All other non-trivial critiques generated by the system were counted. The results are shown in Table 1.

Group	*Fresh*	*Bus*	*ESL*	*Prof*
# of Essays	10	10	10	10
# of Sentences	108	116	110	401
Avg. Words per Sentence	16	18	21	22
# of Different Critiques	20	9	23	32
# of Critiques (Total)	36	11	63	158
% Correct	72%	73%	54%	39%
% Correct and Useful	86%	82%	87%	41%

Table 1. Summary of accuracy for non-trivial critiques

The analysis confirmed feedback we have received from users at IBM Research that CRITIQUE is most helpful on straightforward

texts before they are significantly revised. The more polished and almost literary style of the professional essays challenged CRITIQUE's ability to provide generally useful advice. The ESL texts, written by native Arabic, Chinese, and Spanish speakers, were also difficult, containing a large percentage of very ill-formed sentences. This is indicated by the higher number of ***useful*** critiques for this group, although it could be argued that these critiques may not be as useful to users who lack native intuitions about English. For the ESL group, correcting spelling errors first resulted in significantly better grammar-checking performance. This was not true for the other groups. In general, CRITIQUE also appears to be more accurate on texts with a shorter average sentence length.

Conclusions

Based on real experience in the application areas of office environments, publications organizations and educational institutions, CRITIQUE has been developed to a level of apparent usefulness. Acceptable system performance has been achieved through the use of distributed and optimized processing. The system has achieved a high level of robustness and flexibility in most of its aspects, including presentation. The accuracy of the system is currently acceptable for many types of texts and environments, and accuracy continues to improve with exposure in each of the three application areas. CRITIQUE exemplifies a framework for the development of broad-coverage, large-scale natural language text processing systems.

Acknowledgements

We would like to thank Professor Charles Smith and his colleagues from Colorado State University, who provided the student essays used in the analysis of accuracy, as well as valuable feedback about CRITIQUE. We also express sincere thanks to George Heidorn for his comments and guidance in the preparation of this paper.

References

Binot, Jean-Louis and Karen Jensen. 1987. "A semantic expert using an online standard dictionary." *Proceedings of IJCAI-87*, Milan, Italy, August 1987.

Heidorn, George E. 1982. "Experience with an Easily Computed Metric for Ranking Alternative Parses." *Proceedings of the 20th Annual Meeting of the Association for Computational Linguistics* , Toronto, Canada, June 1982.

Heidorn, George E., Karen Jensen, Lance Miller, Roy Byrd, and Martin Chodorow. 1982. "The EPISTLE Text-Critiquing System." *IBM Systems Journal, 21, 3*, 1982.

Hendrix, Gary. 1986. "Bringing Natural Language Processing to the Micro-Computer Market: The Story of Q&A." *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, New York, New York, June 1986.

Jensen, Karen, George Heidorn, Lance Miller, and Yael Ravin. 1984. "Parse Fitting and Prose Fixing: Getting a Hold on Ill-formedness." *American Journal of Computational Linguistics, 9, 3-4*, 1984.

Macdonald, Nina H., L.T. Frase, P. Gingrich, and S.A. Keenan. 1982. "The WRITER'S WORKBENCH: Computer aids for text analysis," *IEEE Transactions on Communication (Special Issue on Communication in the Automated Office), 30*, 1982.

Richardson, Stephen D. 1985. "Enhanced Text-Critiquing using a Natural Language Parser." *Proceedings of the Seventh International Conference on Computers and the Humanities*, Provo, Utah, June 1985.

Seybold Publications, Inc. 1984. "Computer Aids for Authors and Editors." *The Seybold Report on Publishing Systems, 13, 10*, 1984.