A PC-BASED INTERFACE FOR AN EXPERT SYSTEM
TO ASSIST WITH PREOPERATIVE ASSESSMENTS

E.L. Hoogendoorn\textsuperscript{1}, K.B. Langton\textsuperscript{1}, N. Solntseff\textsuperscript{2}, and R.B. Haynes\textsuperscript{1}

\textsuperscript{1}Health Information Research Unit
Department of Clinical Epidemiology
& Biostatistics
Faculty of Health Sciences
McMaster University Medical Centre
Hamilton, Canada L8N 3Z5

\textsuperscript{2}Department of Computer Science
& Systems
McMaster University
Hamilton, Canada L8S 4K1

Abstract

PREOP (short for preoperative assessment) is a knowledge-based program that can provide physicians with expert-level recommendations concerning the assessment and care of patients who are about to undergo surgery. This paper describes an interface to PREOP designed to provide clinicians with a user-friendly assessment tool. The interface is written in C language for Intel 80x86-based computers with the help of a toolkit of windowing functions. It represents an attempt to implement a mixed-initiative dialogue whereby there is a two-way flow of information between the clinician and the expert system. This makes possible arbitrary correction of user inputs. When complete, PREOP will be able to assess a wider range of risk factors and synthesize more sophisticated recommendations, using hypermedia interface techniques. A goal of this work is to evaluate the impact of artificial intelligence and expert system techniques on medical care.

Introduction

Over the years, the area of medical consultation for surgical patients has received increased attention. The particular type of consultation we are concerned with here is a preoperative assessment of patients who are perceived to be at special risk for surgical complications because of their medical problems. The patient is referred to a medical consultant, usually an internist, who performs an examination to determine if the patient is in acceptable condition for surgery; reduce the risks associated with the specific surgery type and the anesthetics used; and, improve rehabilitation after surgery. To perform these tasks, a consultant has to identify the various factors a patient has which can increase the risk of the operation and make recommendations to reduce the risks.
A typical preoperative assessment consists of a review of: the patient’s medical history; the present condition of the patient; the medications being used by the patient; and, what allergies the patient may have. Often included in the routine investigations are chest radiography (X-ray), electrocardiography (ECG) and laboratory testing of blood glucose, hemoglobin, and electrolytes. The consultant must carefully focus on the specific patient in identifying the patient’s potential risks, while not ordering unnecessary tests nor asking redundant questions.

Clinical researchers have determined criteria for predicting the possibility of postoperative cardiac complications of non-cardiac surgery using cardiac risk factors [1]. We have incorporated these criteria into a knowledge-based computer program, PREOP [2]. At present, the PREOP prototype can do the following: assess cardiac risk factors associated with specific patient variables and the type of operation planned; determine the probability of a patient developing postoperative cardiac complications, using a Bayesian approach; and, recommend when to stop and start various medications. The rules necessary to incorporate this function were developed through close consultation with domain experts and expert reference articles [3, 4]. The PREOP prototype can be used to test the effectiveness of such a tool on the preoperative assessment process.

We have implemented most of PREOP on a VAXstation 3100 using Neuron Data’s expert system shell, Nexpert Object [5]. We chose this environment due to its mixed use of object-oriented and rule-based programming. In clinical practice, the expert system will be used by specialists and trainees in internal medicine, professionals with variable computer skills who are expected to interact with the system by means of portable computers. Because of this feature, the run-time system has to work on industry-compatible computers, as well as presenting an interface that is congenial to this type of user. The goal of the work described below is to investigate techniques for the development of such interfaces.

**Background**

The Nexpert Object system comprises two separate packages; the development system, and the run-time system. The development system includes: several editors (for example, a rule editor, an object editor, and a class editor), a highly graphical user interface, and tools to develop screens and to bridge knowledge-based applications with other products and application programs. The run-time package is a stripped down version of the development system, designed for delivering the complete application. The knowledge base constructed by the development package can be easily ported to a variety of platforms, including: VMS, UNIX, MSDOS, and MacIntosh DOS. Intel 80x86-based machines were used for experimenting with the delivery of our expert system because of their wide availability at McMaster University. The version used in this project was Nexpert Object Development (version 1.1) and NORT (Nexpert Object Run-Time version 1.1, which has now been superseded by Nexpert Forms).

Our early experience with the NORT package revealed that it was not really suited for freely experimenting with interface design, primarily because of its inflexible regime for the presentation of input screens. This rigidity prevented clinical users from backtracking to earlier screens. As a consequence, clinicians were forced to restart a session once data entry mistakes were made. It was decided that the ability to "undo" input is a highly desirable feature of an expert-system run-time interface.
Another idiosyncrasy of NORT is its inability to make use of previously written prompts defined from within the development package. Each run-time program has to be customized with a new set of prompt messages for the user. This is an unnecessary task which increases the probability of programming error. Lastly, in order to customize the NORT interface for specific applications a proprietary graphics package had to be purchased to re-compile and link the code. Therefore, this aspect of the system was outside our control.

**Interface Design**

Neuron Data provides a set of functions, collectively known as the Callable Interface, for construction of interfaces between application programs and the Nexpert inference engine. It is possible to access the knowledge base for any information it contains by means of calls to these functions. The information so obtained can be presented to the user in a manner controlled by the interface designer. We made use of this feature to develop our own interface.

An interface prototype was developed in the C language for Intel 80x86-based computers with the assistance of an interface toolkit [6]. The toolkit’s windowing functions were used to eliminate the need for the previously mentioned proprietary package. The windowing functions can be used to create, present and change text-based windows, display message boxes, activate popup and pulldown menus and handle system errors. There is no mouse interface in the Goodwin toolkit, but we figured that clinicians would find the use of such a pointing device inconvenient for portable use on hospital wards.

The primary criterion for the design of our interface is ease of use by clinicians. More details of the considerations that led to the present design can be found in [7]. At present, the interface is built as a stand-alone program that can be executed with a minimum number of keystrokes so that there should be a short learning time. This version does not adapt to the skill level of the user. This static behaviour is independent of the dynamics of the PREOP knowledge base.

The run-time interface is knowledge-base independent and is capable of processing any knowledge base created with the Nexpert development shell. The interface is geared toward goal-driven operation where the inferencing session begins at a possible goal-state and searches the knowledge space in an attempt to prove the truth of that goal state. The interface must be given the name of the current knowledge base and the goal state. This is accomplished by means of an initialization file created by the knowledge engineer and processed by the interface. The process is transparent to the clinician and occurs at the startup of the interface.

At startup, the clinician is presented with a screen which contains a message announcing that a new session has started, and a pull-down menu bar along the top of the screen. Each menu option is activated by a combination of the ALT key and the corresponding highlighted letter. The menu bar provides options for file manipulation and expert knowledge processing. Menu sub-options currently supported are (1) exit from the system, (2) run an inferencing session, and (3) loading or unloading a knowledge base. Every level of the interface has an associated help screen that can be displayed by pressing the F1 key. Each help screen contains a brief summary of the commands or keystrokes that the client may use at that level. All error messages are signalled by a sound and are displayed in a dialogue window in the center of the screen.
Choice of the Run option from the pulldown menu results in invocation of the Nexpert inference engine that is linked into the interface. The inference engine starts at the goal hypothesis and continues its operation until information is required from the user. The user provides the information through a question-and-answer dialogue controlled by the structure of the knowledge base. Each question specified in the knowledge base is displayed by the interface along with any additional information relevant to that question. This latter information, which may consist of a list of options, format restrictions on the answers and data type constraints, is extracted from the knowledge base by the inference engine and either displayed by the interface or used to ensure the entry of a valid answer. All questions are displayed in a standard format.

Answers to questions may consist of typing an answer or selecting one of the options from the list extracted from the knowledge base. The arrow keys are used to move a highlight bar to the appropriate choice which is then selected by pressing the return key. Pressing any other key opens a window in which the answer is typed. The answer is then checked for validity by ensuring that it is either of the correct data type or exists in the list of choices. If a possible error is detected, the user is warned and given the opportunity to either confirm the answer or make changes.

Each question string, together with the list of available choices, the format, and data type of the answer are recorded to allow the user to review the information at any time during the inferencing processing. Thus, the clinician is able to see that the information entered is correct and that the knowledge processing is taking the proper course. This is a feature not available with the Nexpert Forms and is a step toward providing an accurate model of the interaction between an expert and client [8].

There should be two-way flow of information between an expert system and its user in the course of an inferencing session each responding to the information provided by the other. Such a mixed-initiative dialogue [9, 10, 11] is difficult to incorporate into an interface independently of the structure of the knowledge base and is presently an important research topic. This dynamic interaction will be facilitated if the ability to change already recorded answers is provided. It should be noted, that this complicates the inferencing process as there may be the need to backtrack and selectively undo large parts of the variable bindings. The capability of arbitrary backtracking has been built into our interface and the whole question of its effects on the subsequent inferencing process is under active investigation.

Each time a new answer is displayed, the clinician has the option to modify the value of earlier answers. This feature is activated by pressing the insert key. An audible tone and an error message are used to notify the clinician of the new state of the interface and to get confirmation of the action. The prompt, list of choices, and answer to the first question in the record is displayed. The client may browse through the list of questions to select the appropriate question and answer pair or may return to the main dialogue screen. When a change is to be made to an answer, the status of the inference engine is updated to reflect the changes and the inferencing process continues from that point. This ability to change values is unlimited during any inferencing session.

The interface is capable of executing external processes specified by the knowledge base and of displaying text files that are specified within the rule base. Future versions will add the ability to display graphic images such as those found in pictorial knowledge bases [12] or created to highlight important points. Nexpert supports the use of hypermedia which would make a valuable addition to a medical interface [13, 14, 15].
Because the interface is still in the prototype stage a large number of enhancements are possible. The range of functions available with the Nexpert Callable Interface allows customization of the interface to meet the needs of users ranging from physicians in a clinical setting, to students attempting to fathom the details of preoperative assessment, to knowledge engineers updating databases relevant to the knowledge base. Developing the interface as a Microsoft Windows 3.0 application will provide the added functionality of the Windows environment as well as access to the many enhancements available with such a graphic user interface.

Discussion and Conclusions

The current implementation of PREOP works well within its limited domain of knowledge. Our techniques have proved to be adequate for the construction of interfaces which allow the user to edit data that have been entered at an earlier stage. However, it has been found that backtracking to undo previous information can have unpredictable effects on subsequent inferencing. Although satisfactory for small knowledge bases, it may prove to require too much re-querying of the user to be successful for very large knowledge bases. Research has started on identifying methods to allow for dynamic knowledge-base restructuring following an arbitrary change in knowledge.

As the system grows, we expect its expanded scope to enhance its usefulness. We are also experimenting with other intelligent interface design techniques [16] such as natural language processing and cooperative man-machine dialogue strategies which we hope to incorporate into our system. A result of this work could be better medical care for surgery patients.

Acknowledgments

This work was supported by grants from the B.C. Decker Health Information Research Fund, The Medical Research Council of Canada, and Digital Equipment Corporation of Canada.

References


