# Validation of Expert System Performance

Robert M. O'Keefe
Osman Balci
Eric P. Smith

TR 86-37

# VALIDATION OF EXPERT SYSTEM PERFORMANCE

Robert M. O'Keefe, Osman Balci and Eric P. Smith
Virginia Tech

November 18, 1986

**Abstract**

Most definitions of an expert system include some reference to the ability of the system to perform at a level close to human expert performance. Yet the validation of expert systems, that is, the testing of systems so as to ascertain that they achieve an acceptable level of performance, has (with a few exceptions) been ad-hoc, informal, and in some cases of dubious value.

This paper attempts to establish validation as an important concern in expert systems research and development. The problems in validating an expert system are discussed, and a number of methods for validating expert systems, both qualitative and quantitative, are presented.

## 1 Introduction

Most definitions of an expert system (ES) include some reference to the ability of the system to perform at a level close to human expert performance. Yet the validation of expert systems, that is, the testing of systems so as to ascertain that they achieve an acceptable level of performance, has (with a few exceptions) been ad-hoc, informal, and in some cases of dubious value.

Typically, the performance of an ES has been validated by running a number of test cases through the system, and comparing the "result" (i.e., the classification, final certainty factors, the advice given, or whatever) from the system against either known results or expert opinion. A percentage is calculated for the success rate of the system, and subjective judgement is used to both analyse this and explain the failure of the ES where its result was in contradiction to the known result or expert opinion. Examples of this approach span from an early validation of MYCIN [1], to a recently reported validation of a chest pain diagnosis system called EMERGE [2]. This simple approach presents a number of problems. The final percentage obtained is a function of the choice of test cases, and its accuracy is a function of the number of test cases. Where the system is compared against the expert on whose knowledge the system is built, as happened with PROSPECTOR [3], the value of the so-called validation is dubious.

As many developed expert systems have been research prototypes, the purpose of the validation has often been to qualitatively measure system performance (e.g., see Miller et al. [4] on the medical diagnosis system INTERNIST-1), or validation has simply been part of an overall evaluation aimed at assessing the value of an ES to a particular domain (e.g., see Kulikowski and Weiss [5] on the medical diagnosis system CASNET, or Hansen and Messier [6] on the auditing ES EDP-XPERT). However, the performance of expert systems that are to be used on a regular basis, particularly in critical areas, must be validated very carefully. Thus an increase in the use of formal validation methods can be seen in the development of some implemented systems. Both a later validation of MYCIN [7], and a validation of the chemotherapy adviser ONCOCIN [8], like MYCIN developed at Stanford, used formal methods backed up by statistical tests. The performance of the VAX configuration system R1/XCON also underwent some elements of formal validation (see Bachant and McDermott [9], and the discussion in Gaschnig et al. [10]).

## 1.1 Verification, Validation and Evaluation

It should be noted that here we are only concerned with the *validation* of performance. Typically, this is part of the broader area of *evaluation*, which seeks to assess the overall value of an ES. In addition to exhibiting an acceptable level of performance, an ES should be useable, efficient, and cost-effective. Validation is the cornerstone of evaluation, since, for example, a highly effecient implementation of an invalid system is useless.

It can often be difficult to separate validation of performance from other aspects of evaluation. For instance, testing can be difficult if users balk at using the system due to lack of human factor considerations, e.g., a poorly designed interface. For this reason, it is important that at the outset of development it is decided whether or not to separate validation. In many instances, particularly ground breaking applications, a policy of overall evaluation may seem more relevant, especially if at the start it is determined that a reasonable level of performance will be difficult to obtain. For a discussion of evaluation, see Gaschnig et al. [10].

Validation is often confused with *verification*. Simply stated, verification refers to building the system right (i.e., substantiating that the system is a correct implementation of the specification), whereas validation refers to building the right system (i.e., substantiating that the system performs with an acceptable level of accuracy). In modeling studies, it is well to remember the dictum that nobody solves *the* problem; rather, everybody solves the model that he has constructed of the problem. Since an ES is a model of human reasoning and knowledge, its level of representativeness must be justified through the process of validation.

The purpose of this paper is twofold. First, we wish to establish validation, and evaluation, as an important concern in ES research and development. Second, we present both qualitative and quantitative methods of formal validation, some taken from other areas of computer-based model validation, that can be applied to expert systems. The work presented here draws upon our experience in developing expert systems (some of which is presented in O'Keefe et al. [11]) and using validation methods in a number of areas of computer-based modeling.

In the next section of the paper we cover the problems that are encountered in trying to validate ES performance. This is followed by a discussion of a number of basic concepts fundamental to validation, a review of some appropriate qualitative methods, and a discussion on the use of quantitative methods. The paper finishes with a number of conclusions regarding the application of validation methods.

## 2 The Problems of Validating an Expert System

We believe that there are seven major problems that are encountered in trying to validate ES performance: (1) what to validate, (2) what to validate against, (3) what to validate with, (4) when to validate, (5) how to control the cost of validation, (6) how to control bias, and (7) how to cope with multiple results. All of these, to a greater or lesser extent, are encountered in trying to validate any computer-based model. In this section, we discuss these problems and provide some guidelines.

### 2.1 What to Validate

As pointed out by Gaschnig et al. [10], it is possible to validate either the final result (often called the conclusion) of the system, any intermediate results that are obtained, the reasoning of the system, or any combination of these three. Chandrasekaran [12] discusses the importance of validating the reasoning process, since a poor reasoning process that provides a correct result can not be "scaled up" to a larger domain of application, or may give a different result when used with an extended knowledge base.

What to validate is intrinsically linked to which stage we are at in the development process. At any stage, if the performance of part of the system can be measured for a given set of inputs, that part should be validated so as to catch errors as early as possible in the development life cycle. Typically, we may wish to concentrate on validating the reasoning process early on in development, and only be concerned with the validity of final results when the knowledge base is more complete.

A further problem is that it may be difficult, and in some cases impossible, to classify the result of an ES as right or wrong. In discussing the validation of CASNET, Kulikowski and Weiss [5] state the following:

> "Classifying conclusions as being merely correct or incorrect is an oversimplification. The program's conclusions are presented not as single unique diagnoses but rather as combinations of judgements about a patient's status."

However, in such cases it is often possible to get experts to classify the results, intermediate results and reasoning into several categories. For instance, Hickam et al. [8] got chemotherapy protocol experts to classify the performance of ONCOCIN as ideal, acceptable, sub-optimal, and unacceptable.

## 2.2 What to Validate Against

As well as being validated against expert performance, in some cases an ES can be validated against known results (e.g., in validating an ES that predicts the financial performance of a company in the upcoming financial year, for past case histories the present financial position of the company will give a known result). As many have pointed out (e.g., Chandrasekaran [12]), it can be unfair to expect an ES to perform at a level close to known results when human experts can not perform at this level. Typically, expert systems should be validated against experts, although where available known results can provide a useful background for validation.

A problem with known results is that previous expert decisions may precipitate the result. Suppose that in the financial system example above, the prediction of performance is used by a bank to decide whether or not to continue financial support of the company.

If an expert had decided that the financial position of company X will be very poor in a years time, and thus implements withdrawal of financial support, the presently known poor financial position of X may in part be due to the previous expert decision.

## 2.3 What to Validate With

In an ideal world, a large number of documented previous cases, representing work from a number of experts on a complete range of problems, will be available for use in validation. Typically, in a less than ideal world, only a small sample will be available, drawn from a single or a few experts. In extreme cases, no test cases will be available.

Further, the "success" of any validation is biased by the choice of test cases. Any used in the development of the system should be discarded, since (supposedly) the system will have already been altered so as to successfully handle the case. In many domains, a large number of situations are fairly standard, and can be handled with limited expertise. For example, suppose we are developing a medical diagnosis system in a domain where 90% of the cases are standard, and the other 10% require considerable skill. A success rate for the ES, when validated against test cases, of 90%, would not inspire confidence in the system.

For a fair cross-sectional validation of the system, the sample of test cases should be randomly selected using stratified sampling, that is, randomly selected within each identifiable type of result. For instance, if an expert classifies a diagnosis as A, B or C, and previous histories indicate that these classifications respectively occur 80%, 15% and 5% of the time, then a collection of 200 test cases should include 160, 30 and 10 instances where A, B and C were respectively the result. In many instances, more detailed validation will require testing the system against a small number of obscure or difficult cases, perhaps cases that even top experts find difficult, and qualitatively assessing how well the system handles these.

The number of test cases used obviously has an effect on the confidence that can be placed upon the validity of the system performance. However, the law of large numbers simply does not apply here. The issue is not the largeness of the number of cases; it is the *coverage* of the test cases. This is how well they reflect the ***input domain***. The ***input domain*** is the population of permissible input [13]. (This should not be confused with the ***application domain*** in which the ES operates.) The larger the size of the input domain, the more difficult it becomes to validate the ES.

If no historic test cases are available, or all of them have been used in developing the system, it may be possible to synthesise test cases by getting experts to randomly create cases. This is problematical, since any set of synthesised cases is unlikely to represent a well stratified sample. Further, experts are unlikely to expend as much time or effort on a synthetic problem as on a real problem, and their reasoning and results may well suffer. However, in some instances, this may well be the only means of providing such a sample of test cases.

## 2.4 When to Validate

There is little agreement on when to validate an ES. In discussing R1/XCON, Bachant and McDermott [9] relate the folly of expecting high performance from a system early in its career, stating that:

> "To expect anything close to perfection during the first few years a system is being used (especially if the task is significantly more than a toy) is probably a very serious mistake."

whereas Buchanan and Shortliffe [14] (p. 695) state that:

> "we believe that high performance is a *sine qua non* for an ES and thus deserves separate evaluation early in a program's evolution."

Although these two statements appear to be contradictory advice, we believe that there is really no contradiction. Typically, an ES must exhibit acceptable performance at some early stage of development. However, the level of acceptance may be very different for differing systems. With a research prototype, medium performance, and indications that the basic approach is correct, may be acceptable. As the system is extended, further validation may be necessary, but such validation should concentrate on validating the extensions, indicating areas that may need improvement or fine tuning. With non-critical applications, such as R1/XCON, it is possible to put the system in the field and validate the system as it is used. In critical applications, where life (e.g., some medical diagnosis applications) or large amounts of money (e.g., some manufacturing applications) is at stake, field testing is usually impossible (although in certain instances it may be possible to run the system in parallel with the present manual system).

## 2.5 How to Control the Cost of Validation

Validation can be time-consuming and expensive. Assessing the amount of money and effort that should be applied to validation is a difficult task.

We believe that the cost of validation can be controlled by designing formal validation methods that are integrated within the development process. The value of validation is obviously directly linked to the value of the system to its users, and the risk involved in using a poorly validated system. These ideas are developed further in a subsequent section.

## 2.6 How to Control Bias

An ES may be validated against an expert whose expertise is biased when compared to other experts. When judging ES performance, an expert biased against the introduction of computer-based systems may give similarly biased opinions, unfairly assessing the ES. Selecting the set of test cases so as to guarantee good performance by the ES is an example of developer bias.

Expert bias can be controlled by the use of blinded evaluation and cross checking, both used in the ONCOCIN validation [8]. When judging performance, experts should not be able to distinguish between human expert and program performance. Bias between experts can be checked for with a number of statistical tests, as will be discussed later. Developer bias can only be controlled by honest professionalism.

### 2.7 How to Cope with Multiple Results

A serious problem exists with the validation of an ES with multiple results, refered to as the *multiple response* problem. It is not appropriate to test the validity of a multivariate response ES by testing the validity separately for each of the response variables. (See Shannon [15] (p. 229) for an illustration of this problem in validation of simulation models). A multivariate approach must be used to incorporate the correlation among the results and to test for the *overall* validity of the ES.

For example, in a medical diagnosis system that prescribes appropriate drug treatment for patients, prescription of two types of drug may be valid if each is separately considered as a treatment, yet the combination of the drugs may be unacceptable, and hence the *overall* response of the system is invalid.

## 3 Some Basic Concepts

In this section, we introduce a number of basic concepts in the validation of expert systems, namely, acceptable range of performance, input domain, test of validity, formality of validation, and ES builder's/ES user's risk.

### 3.1 Acceptable Range of Performance

The concept of validation should not be considered as a binary decision variable where the ES is *absolutely* valid or *absolutely* invalid. Since an ES is a representation or an abstraction of reality, perfect performance can not be expected. The level of performance that is acceptable to the users is called the *acceptable range of performance*, and should be specified at some stage of development. In some cases, an acceptable range of performance may be specified by a third party, a government agency, or the project sponsor.

Suppose that an ES produces a classification A, B, C or D. It may be possible to specify a acceptable range of performance in terms of the occurence of each result. For instance, it may be decided that the system should always correctly classify A and B, but that some variation in classifying C and D is acceptable. Further, the probability of a particular incorrect classification may be vital. For example, it may be vital that the system never gives a result of B when the known result is A, i.e., $Prob(B \mid A)$ should be shown to be zero.

Frequently, the acceptable range of performance will reflect the ability to perform at a level equivalent to human expertise. Bachant and McDermott [9] pointed out that:

> "the people who used R1 did not demand more of it than of its human predecessors."

Thus it may be possible to specify an acceptable range of performance with regard to the performance of the human experts that the system models.

## 3.2 Input Domain

An ES should be developed for a specific purpose or application and its adequacy or validity should be assessed only in terms of that purpose with regard to a prescribed *input domain*. An ES may be valid for one input domain, and completely absurd for another.

## 3.3 Test of Validity

It is important to realise that an ES can only be tested for validity under a prescribed *input domain* and for an *acceptable range of performance* related to the purpose for which the ES is intended.

## 3.4 Formality of Validation

The application of validation can range from formal to informal. Formal validation requires establishing when validation should occur within the development life cycle, the identification of validation methods, the specification of the input domain and the level of acceptance,

|  |  | State of the ES |  |
|---|---|---|---|
|  |  | ES is Valid | ES is Invalid |
| Action | Accept as Valid | Correct Decision | ES User's Risk (Type II Error) |
|  | Declare Invalid | ES Builder's Risk (Type I Error) | Correct Decision |

Figure 1: Types of Risk in Validation

and (where appropriate) the relevant application of statistical techniques. Informal validation is typically done at the end of development, is often an after-thought, and employs ad-hoc methods.

As would be expected, most validation processes are somewhere between formal and informal. Yet it is generally accepted that validation is frequently too informal, and that developers do not consider their approach to validation early enough, and hence do not build it into the development life cycle.

## 3.5 ES Builder's/ES User's Risk

Testing validity can result in four possible outcomes as depicted in Figure 1. Two of these outcomes are called the type I and type II errors. Type I error is commited if the validity of the ES is rejected when in fact it is sufficiently valid. Type II error is commited if the validity of the ES is accepted when in fact it is invalid. The probability of type I error is called the *ES Builder's Risk* and the probability of Type II error is called the *ES User's Risk*. (See Balci and Sargent [16] for a quantification of these risks in using statistical hypothesis testing for the validation of simulation models.)

The major consequence of commiting type I error is that the cost of developing the ES is going to increase unnecessarily. The additional cost may not be negligible; for example, it may precipitate the abandonment of a system that has cost a considerable amount of effort to develop. On the other hand, the consequences of commiting type II error can be very dramatic. For example, a medical diagnosis system may incorrectly diagnose a patient's illness, with the consequence that the patient suffers through mistreatment. Therefore, in critical applications, it is essential that we minimize the User's Risk as much as possible. Objective consideration of the two types of risk, their relative importance, and the cost if each should occur, will help to provide a basis for the design of the validation process.

## 4 Qualitative Validation

Qualitative approaches to validation employ subjective comparisons of performance. This does not imply that they are informal; it is possible to design a highly formal qualitative validation process. Qualitative methods can be combined with quantitative ones, where appropriate. If ES responses can somehow be quantified (ie., expressed in terms of numbers), then we can employ quantitative (statistical) techniques.

In this section we review seven common qualitative approaches to validation, some of which have been used in validating expert system performance.

### 4.1 Face validation

Face validation is useful as a preliminary approach to validation. The project team members, potential users of the ES, people knowledgeable about the application domain, using their knowledge and intuition, subjectively compare ES performance against human expert performance. The results obtained from an ES running under a given set of test cases are assessed "at face value" with regard to a prescribed acceptable range of performance.

As discussed by McDermott [17], R1/XCON was validated by a group of six experts reviewing its performance on fifty orders. Perceived mistakes were rectified prior to implementation.

## 4.2 Predictive Validation

Predictive validation requires using historic test cases, and either known results or measures of human expert performance on those cases. The ES is driven by past input data from the test cases, and its results are compared with the corresponding results, either known or obtained from the human expert. As discussed previously, a number of reported ES validations, e.g., Hudson et al. [2], have used this approach.

## 4.3 Turing Tests

Turing tests aim to validate an ES against human experts, by getting experts to evaluate the performance of other experts and the ES without knowledge of who is performing. The assessments of system performance can then be compared with the assessments of human performance. This process of blind evaluation has the advantageous side effect of eliminating any anti or pro computer bias. If the assessments can be objectively measured (e.g., experts can conclude that a test case was handled correctly or incorrectly, or assess performance to be expert, good, fair, or poor), then statistical techniques can be used to test for variation between the ES and human experts, and consistency between experts.

Chandrasekaran [12] has argued for using Turing tests in validating medical expert systems. Such tests have been used in validating both MYCIN [7] and ONCOCIN [8].

## 4.4 Field Tests

Fields tests involve placing a prototype version of the ES "in the field", and then catching perceived errors in performance as they occur. From the developers point of view, this has two considerable advantages. First, the burden of testing is placed upon the users. Second, the acceptable range of performance is obtained implicitly, since users may cease to report problems when an acceptable range of performance is reached. (This may, of course, backfire, with users reporting any minor problem that occurs for the duration of the existence of the system).

Field testing is only possible in non-critical applications, where users can assess the correctness of the ES performance. Subsequent to its implementation, R1/XCON underwent considerable field testing, as discussed by Bachant and McDermott [9]. With R1/XCON, an incorrect result is easy to observe, since the VAX configuration will either not fit together or not work.

## 4.5 Subsystem Validation

Subsystem validation requires the decomposition of the ES into subsystems, where the performance of each can be observed under a given set of input data. In this approach, each subsystem is validated one at a time as they are developed.

Subsystem validation has three significant advantages: (1) validation is incorporated within the development life cycle and carried out along with the development, (2) it is much easier to validate subsystems since they are less complex and more manageable, and (3) error detection is much easier since they are localized. The disadvantages of subsystem validation are that it may not be possible to observe the input-output behavior of a subsystem, and successfully validating each subsystem does not imply the overall validity of the whole ES since error tolerances can accumulate to be significant in the overall performance of the ES.

In some instances, it may be possible to identify a subsystem where all possible outputs can be generated for all possible inputs. This can occur in production rule systems, where a group of rules may apply crisp logical inference on a limited discrete input domain. If user judgements are gathered on a continuous scale, and used as certainty factors, it is possible to simulate the performance of a subsystem over the range of permissible input. For an example of this, see Langlotz et al. [18].

## 4.6 Sensitivity Analysis

Sensitivity analysis is performed by systematically changing the values of the ES input variables and parameters over some range of interest and observing the effect upon the performance of the ES. Suppose we have a system that gives complete satisfaction (i.e., final result, intermediate results and reasoning are all assessed to be sufficiently expert) when dealing with case C. If C uses inputs (data, user judgements etc.) $i_1, i_2, ..., i_n$, then a sensitivity analysis validation would involve altering each input, and assessing the change in performance of the ES. For instance, if $i_3$ is "the temperature of the patient", and our satisfactory consultation case included a temperature of 101 degrees, and further we know that for this case temperature should have no effect on diagnosis, then altering $i_3$ to any other value, while leaving $i_1$, $i_2$ and $i_4, i_5, ..., i_n$ as before, should not alter the final result, intermediate results, or perhaps even the reasoning process.

To our knowledge, sensitivity analysis has not been used by expert system developers as a validation method, or at least not explicitly used. It may prove to be the most powerful qualitative method available; it is especially useful where few or no test cases are available. It is also highly appropriate for systems which use uncertainty measures, and require users to provide judgements for premise uncertainty, since these can be altered as desired, and the effect on intermediate and final uncertainty measures can be examined.

## 4.7 Visual Interaction

Visual interactive validation involves providing a visual animation of the workings of the ES, and allowing experts to interact with it, altering parameters as desired. In essence, this can be viewed as an environment for interactive face validation, subsystem validation, and sensitivity analysis. It has been very successfully employed in the validation of Operations Research models, particularly discrete-event simulations [19].

This approach to validation may have some applicability to expert systems, particularly with the appearance of a number of graphical interfaces to knowledge-based systems (e.g., see Richer and Clancey [20] on GUIDON-WATCH). Experts can watch the reasoning process, the access of rules, the propergation of uncertainty etc., and hence validate the reasoning process.

## 5 Quantitative Validation

Quantitative approaches to validation employ statistical techniques for the comparison of ES performance against either test cases or human experts. Here we present, and discuss the applicability, of three types of quantitative methods : (1) paired $t$-tests, (2) Hotelling's one-sample $T^2$ test, and (3) simultaneous confidence intervals. This is followed by some discussion on measuring consistency between experts, an important issue when performing validation against multiple experts.

Quantitative validation methods generally fall into two categories. Either a *confidence interval* for one or more measures is constructed, and subjectively compared against an acceptable range of performance, or a formal *hypothesis test* is used to compare measurements against a predetermined acceptable range of performance, where the hypotheses are :-

$\mathbf{H_0}$ : The ES is valid for the acceptable range of performance under the prescribed input domain.

$\mathbf{H_1}$ : The ES is invalid for the acceptable range of performance under the prescribed input domain.

### 5.1 Paired $t$-Tests

As has been seen, producing a single proportion as a measure of performance is of limited value. A far more appropriate method is to use a paired $t$-test to compare the difference between observed results. For the final result from the ES, the difference between ES performance, and human expert performance or known results, is measured. We construct an interval $D_i$ where $D_i = X_i - Y_i$, $X_i$ are results from the ES, and $Y_i$ are either known results or results from human expert performance. Notice that this can cover many types of result. If a final judgement is produced on a continuous scale, for example -5 to +5 as in PROSPECTOR, then $X_i$ can be the system's judgement, and $Y_i$ can be the human expert's judgement. If both known results and expert performance is known, we could (for example) get a third-party expert to measure the performance of each on a scale of, for instance, 1 to 10, under blind evaluation, and thus $X_i$ and $Y_i$ will be the absolute performance of the system and the expert, respectively.

For the differences $D_i$ a confidence interval can be produced thus

$$\bar{d} \pm t_{n-1,\alpha/2} S_d / \sqrt{n}$$

where $\bar{d}$ is the mean difference, $S_d$ the standard deviation, and $t_{n-1,\alpha/2}$ is the value from the $t$ distribution with $n$ degrees of freedom, $n$ being the number of test cases. If zero lies in the confidence interval, than we can accept $H_0$.

## 5.2 Hotelling's One-Sample T$^2$ Test

A previous section discussed the problems of multivariate responses. Whilst a paired $t$-test is appropriate where a *single final result* is obtained from the ES, simultaneously applying a paired $t$-test to a number of final results is inappropriate, since performances on each measure can be expected to be correlated. In this case, Hotelling's one-sample T$^2$ test should be used.

Consider the validation of an ES against a human expert performance. Exactly the same input is given to the ES and to the human expert. Assuming that we have k responses (measureable results) as the expected output, we determine the differences between the corresponding k paired responses. Repeating this for different input values we construct k vectors of differences, one for each response. Then, the one-sample T$^2$ test is used to test if the means of the difference vectors are significantly different from zero simultaneously (or jointly). (See Balci and Sargent [21] about how this test is used in the validation of multivariate response trace-driven simulation models.)

## 5.3 Simultaneous Confidence Intervals

Simultaneous confidence intervals or joint confidence regions can be constructed for the differences of the paired responses to validate a multivariate response ES. Usually, the confidence intervals or regions constructed are compared with a prescribed acceptable range of performance. (See Balci and Sargent [22] about how this approach is used in the validation of simulation models.)

## 5.4 Measures of Consistency

If an ES is to be compared against a number of experts, or multiple experts are going to be compared against the system in a Turing test, then a major consideration is the consistency between experts (often called *inter-observer reliability*).

One commonly used measure of consistency is the interclass correlation coefficient, which has been used, for instance, to evaluate reliability between legal judges. If a judgement by expert $i$ on text case $j$ is denoted $Y_{ij}$, then a model for expert reliability is $Y_{ij} = U + E_i + \xi_{ij}$, where $U$ is the mean rating across all experts, $E_i$ is the effect of the $i$th expert (ie., deviation from the mean $U$), and $\xi_{ij}$ is the error term. From this model, a correlation co-efficient can be produced and tested. If $Y_{ij}$ is a categorical variable (for example, expert, good, fair or poor), rather than a continuous variable, then the kappa statistic can be used to measure reliability [23].

A related statistic suggested by Williams [24], compares the joint agreement of several experts with that of the system. This is appropriate in Turing tests where the judgement of multiple experts is agregated, for example, where the ES performs a task normally performed by a number of related experts. Other tests, including confidence interval approaches, are discussed in Fleiss [25].

## 6 Conclusions

As was once the case with knowledge aquisition, validation of performance is seen as a "black art". Yet few would disagree with the statement that an ES should exhibit an acceptable and reliable level of performance. Validation, and evaluation, is an important concern for most ES developers, that requires considerably more attention then it presently receives.

Throughout this paper, we have made a number of points regarding the problems of, and approaches to, validating ES performance. We will reiterate some of the more important ones here:

1. A system can only be validated against an *acceptable range of performance*, for a prescribed *input domain*. Previous human performance can give an indication of the acceptable range.

2. Validation should be built into the development cycle. It is often necessary to carry out a cross-sectional validation of performance prior to implementation, and specific validation tests as the system evolves after implementation.

3. It is important to consider the risks involved in using a system that is actually invalid (ES user's risk), relative to the risks involved in not using a system that is actually valid (ES builder's risk).

4. Choose an appropriate qualitative method. Field testing may be acceptable for non-critical applications. Turing tests are useful for comparing the system against experts in a blinded evaluation, and can avoid pro or anti computer bias. Subsystem validation and sensitivity analysis are useful for validating specific areas of concern.

5. Use quantitative methods where applicable. Use them as informatively as possible, for instance, produce confidence intervals rather than single point estimates. Be aware of the multiple response problem, and use appropriate multivariate techniques.

However, the bulk of this paper has been necessarily *descriptive*. What is needed by ES developers is a methodology that is *prescriptive*, ie., explains how to validate expert systems under certain conditions (eg., consultative applications, real-time critical applications) and certain constriants (eg., money, development time). Yet experience in ES validation is, at the present time, limited. A methodology, or methodologies, will only evolve in the light of future collective experience and critical appraisal of that experience.

**Acknowledgements**

We are grateful to Dr. John Roach for providing a number of useful comments.

## References

1. V.L. Yu, B.G. Buchanan, E.H. Shortliffe, S.M. Wraith, R. Davis, A.C. Scott and S.N. Cohen, "Evaluating the Performance of a Computer-Based Consultant", *Computer Programs in Biomedicine* Vol. **9**, 1979, pp. 95-102.

2. D.L. Hudson, M.E. Cohen, P.C. Deedwania and P.E. Watson "Prospective Analysis of EMERGE, an Expert System for Chest Pain Analysis", In *IEEE Computers in Cardiology*, IEEE, Silver Springs, MD, 1984, pp. 19-24.

3. J. Gaschnig "Preliminary Performance Analysis of the PROSPECTOR Consultant System for Mineral Exploration", *Proc. Sixth Int'l Joint Conf. Artificial Intelligence*, 1979, pp. 308-310.

4. R.A. Miller, H.E. Pople and J.D. Myers, "INTERNIST-1, An Experimental Computer-Based Diagnostic Consultant for General Internal Medicine", *The New England Journal of Medicine*, Vol. **307**, 1982, pp. 468-476.

5. C.A. Kulikowski and S.H. Weiss, " Representation of Expert Knowledge for Consultation: The CASNET and EXPERT Projects", in P. Szolovits, ed., *Artificial Intelligence in Medicine*, AAAS Selected Symposium Series 51, Westview Press, Boulder, CO, 1982 pp. 21-56.

6. J.V. Hansen and W.F. Messier, "A Preliminary Investigation of EDP-XPERT", Accounting Research Center Working Paper 85-6, School of Accounting, University of Florida, 1985.

7. V.L. Yu, L.M. Fagan, S.M. Wraith, W.J. Clancey, A.C. Scott, J. Hannigan, R.L. Blum, B.G. Buchanan and S.N. Cohen, "Antimicrobial Selection by a Computer", *Journal of the American Medical Association*, Vol. **242**, 1979, pp. 1279-1282.

8. D.H. Hickam, E.H. Shortliffe, M.B. Bischoff, A.C. Scott and C.D. Jacobs,"The Treatment Advice of a Computer-Based Cancer Chemotherapy Protocol Advisor", *Annals of Internal Medicine*, Vol. **103**, 1985, pp. 928-936.

9. J. Bachant and J. McDermott, "R1 Revisited: Four Years in the Trenches", *AI Magazine*, Vol. **5**, 1984, pp. 21-32.

10. J. Gaschnig, P. Klahr, H. Pople, E. Shortliffe and A. Terry, "Evaluation of Expert Systems: Issues and Case Studies", in F. Hayes-Roth, D.A. Waterman and D.B. Lenat, eds., *Building Expert Systems*, Addison-Wesley, Reading, MA, 1983, pp.241-280.

11. R.M. O'Keefe, V. Belton and T. Ball, "Experiences with Using Expert Systems in O.R.", *Journal of the Operational Research Society*, Vol. **37**, 1986, pp. 657-668.

12. B. Chandrasekaran, "On Evaluating AI Systems for Medical Diagnosis", *AI Magazine*, Vol. **4**, 1983, pp. 34-37.

13. W.C. Hetzel, "Principles of Computer Program Testing", in W.C. Hetzel, ed., *Program Test Methods*, Prentice-Hall, Englewood Cliffs, NJ, 1973, pp.17-28.

14. B.G. Buchanan and E.H. Shortliffe, *Rule-Based Systems: the MYCIN Experiments of the Stanford Heuristic Programming Project*, Addison-Wesley, Reading, MA, 1984.

15. R.E. Shannon, *Systems Simulation: The Art and Science*, Prentice-Hall, Englewood Cliffs, NJ, 1975.

16. O. Balci and R.G. Sargent, "A Methodology for Cost-Risk Analysis in the Statistical Validation of Simulation Models", *Communications of the ACM*, Vol. **24**, 1981, pp.190-197.

17. J. McDermott, "R1: The Formative Years", *AI Magazine*, Vol. **2**, 1981, pp. 21-29.

18. C.P. Langlotz, E.H. Shortliffe and L.M. Fagan, "Using Decision Theory to Justify Heuristics", *Proceedings of AAAI '86*, 1986, pp. 215-219.

19. P.C. Bell, "Visual Interactive Modelling In Operational Research: Successes and Opportunities", *Journal of the Operational Research Society*, Vol. **36**, 1985, pp. 975-982.

20. M.H. Richer and W.J. Clancey, "GUIDON-WATCH: A Graphic Interface for Viewing a Knowledge-Based System", *IEEE Computer Graphics and Applications*, Vol. **5**, 1985, pp. 51-64.

21. O. Balci and R.G. Sargent, "Validation of Multivariate Response Trace-Driven Simulation Models", in A.K. Agrawala and S.K. Tripathi, eds., *Performance '83*, North-Holland, Amsterdam, 1983, pp.309-323.

22. O. Balci and R.G. Sargent, "Validation of Simulation Models Via Simultaneous Confidence Intervals", *American Journal of Mathematical and Management Sciences*, Vol. **4**, 1984, pp. 375-406.

23. J. Cohen, "Weighted Kappa: Nominal Scale Agreement with Provision for Scaled Disagreement or Partial Credit", *Psychology Bulletin*, Vol. **70**, 1968, pp. 213-220.

24. G.W. Williams, "Comparing the Joint Agreement of Several Raters With Another Rater", *Biometrics*, Vol. **32**, 1976, pp. 619-627.

25. J.L. Fleiss, *Statistical Methods for Rates and Proportions*, John Wiley and Sons, New York, NY, 1981.