Gray Box Robustness Testing of Rule Systems

Joachim Baumeister, Jürgen Bregenzer and Frank Puppe

Department of Computer Science
University of Würzburg, 97074 Würzburg, Germany
Phone: +49 931 888-6740, Fax: +49 931 888-6732
email: {baumeister, bregenzer, puppe}@informatik.uni-wuerzburg.de

Abstract. Due to their simple and intuitive manner rules are often used for the implementation of intelligent systems. Besides general methods for the verification and validation of rule systems there exists only little research on the evaluation of their robustness with respect to faulty user inputs or partially incorrect rules. This paper introduces a gray box approach for testing the robustness of rule systems, thus including a preceding analysis of the utilized inputs and the application of background knowledge. The practicability of the approach is demonstrated by a case study.

1 Introduction

Validation and verification are two key issues for evaluating the real world practicability of intelligent systems. In the past, an extensive amount of research was undertaken for the evaluation of rule-based representations of such systems, e.g., [1–3]. Alternative knowledge representations like Bayesian networks are also suitable and often more precise for the formalization of domain knowledge. However, using rules is still very popular due to their compact and intuitive manner, e.g., currently the ontology layer of the Semantic Web stack is extended by an appropriate rule representation [4].

Whereas the correct behavior of rule bases (validation) and the correct implementation of rules bases (verification) have been investigated in much detail, there is only little research available when such systems are applied in noisy environments. According to Groot et al. [5] we call such an evaluation task *robustness testing*. In the context of intelligence enriched services on the web, e.g. Semantic Web applications, the robustness is very important, since these services are usually provided for general users.

Robustness testing evaluates the correct behavior of the system with respect to either incorrectly entered input or partially occurring errors in the rule base. Incorrect inputs are the result of user mistakes that can be explained by uncertainty or ignorance of the user when providing the answer to a particular question. A partially incorrect rule base is often the result of a biased or incomplete knowledge acquisition. With robustness testing such effects can be simulated by so called *torture tests*, that gradually decrease the quality of the inputs or the quality of the available knowledge. In general, torture tests are an extension of the well-known empirical testing method. With empirical testing a collection of previously solved and correct test cases is given to the knowledge system as input. Then, for each test case the solutions derived by the knowledge system are compared with the solutions already stored in the cases. Typically, measures like the precision, the recall or the E-measure are used for a quantitative comparison of the two solution sets. Torture tests run a series of empirical tests by degrading the quality of settings, e.g. by gradually degrading the input quality or by gradually worsening the quality of the rule base.

In this paper, we present torture tests as a gray box testing technique thus extending the basic work by Groot et al. [5]. We motivate that a sound implementation of robustness testing should be preceded by a thorough analysis of the applied case base and rule base, respectively. With the results of such an analysis additional background knowledge can be applied yielding a reliable simulation of typical system usages.

The rest of the paper is organized as follows: In Section 2 we introduce the basic notions for defining a rule-based system and sketch general measures for evaluating and comparing the robustness of intelligent systems. In Section 3 we introduce the phases of a degradation study, i.e., the implementation of a robustness testing. Furthermore, we define measures for the analysis of the case base and rule base, and we discuss implications that can be drawn from such an analysis. A case study with two rule-based consultation systems is presented in Section 4. The paper is concluded with a summary and an outlook in Section 5.

2 Measuring the Quality of Rule-Based Systems

We consider a rule-based system as a *consultation system*, i.e., a system deriving suitable solutions for a stated problem description. More formally, the input and output of such a system is defined as follows.

Definition 1 (Input and Output). Let Ω_{obs} be the (universal) set of observable *input values*. A tuple $f \in \Omega_{obs}$ with $f = a : v$ is often called a finding, where $a \in \Omega_a$ is an input (attribute) and $v \in dom(a)$ is an assignable value.
Let Ω_{sol} be the universe set of (boolean) *output values*, i.e. solutions derivable by the knowledge system.

A problem solving session is represented by a case containing the specified input and (derived) output values for a given problem.

Definition 2 (Case). A case c is defined as a tuple $c = (\, OBS_c, SOL_c \,)$, where $OBS_c \subseteq \Omega_{obs}$ is the *problem description* of the case, i.e., the observed input values of the case c; $OBS_c = \{f_{1,c}, \dots, f_{n,c}\}$. The set $SOL_c \subseteq \Omega_{sol}$ contains the (derived) outputs of case c.

A *test suite* is a collection of (test) cases that is used for robustness testing. In general, we define a quality function for a given knowledge system and a collection of test cases as follows.

Definition 3 (Rule Base). A *rule r* is defined as $r : c_r \rightarrow a_r$, where the rule condition c_r is a combination of conjunctions/disjunctions of findings $f \in \Omega_{obs}$, and a_r is the rule action (consequent) of r. In the following, we see that a_r will be used for deriving outputs or for implementing a dialog strategy. A rule base $\mathcal{R}$ is defined as a collection of rules defined in the input and output space.

Examples of rule types are given in Section 3.1 for rule base properties.

Definition 4 (Quality Function). Let C be the universe of test cases, i.e., containing all possible and reasonable combinations of input values, and let $\mathcal{R}$ be the rule base. Then, $q : C \times \mathcal{R} \to [0, 1]$ is a *quality function* comparing the expected solution documented in a case c and the solution of c derived by $\mathcal{R}$.

Examples for a quality function are the *precision*, the *recall* or applications of the *E-measure*, e.g., the F-measure. In the following, we will discuss some properties of knowledge systems with respect to the quality of its output. When the input quality is degraded, then the system should show a monotonically degrading output quality, i.e., no fluctuating output quality. In consequence, the output quality of the system is more predictable. More formally we define the monotonic derivation quality of a system as follows.

Definition 5 (Monotonic Derivation Quality). For a rule base $\mathcal{R}$ let $C = (c_1, \ldots, c_n)$ be a sequence of cases sorted according to their input quality in ascending order; further let q be a quality function. Then the system shows a *monotonic derivation quality*, if $q(c_i, \mathcal{R}) \leq q(c_{i+1}, \mathcal{R})$ for all $1 \leq i \leq n$.

This criterion is a necessary requirement for any knowledge system, that should be considered to be *robust*. Groot et al. [5] additionally introduce measures for comparing two knowledge systems discussing their robustness: the quality value and the rate of quality change. We briefly describe them in the following.

Definition 6 (Quality Value). For two rule bases $\mathcal{R}_1$ and $\mathcal{R}_2$ we say that $\mathcal{R}_1$ is more robust than $\mathcal{R}_2$ for a given quality function q, if for any test case c the output quality of $\mathcal{R}_1$ is higher than the output quality of $\mathcal{R}_2$, i.e., $q(c, \mathcal{R}_1) > q(c, \mathcal{R}_2)$.

Whilst the *quality value* considers the isolated behavior of the system for each test case, the *rate of quality change* emphasizes the comparison of dynamic behavior of the systems.

Definition 7 (Rate of Quality Change). For two rule bases $\mathcal{R}_1$ and $\mathcal{R}_2$ and a quality function q, we say that $\mathcal{R}_1$ is more robust than $\mathcal{R}_2$, if for any qualitatively ordered sequence of test cases the average quality of the output of $\mathcal{R}_1$ decreases more slowly than for $\mathcal{R}_2$. The average output quality is computed by a series of degradation sequences (see Section 3).

The measures presented so far are useful for comparing different aspects of the robustness of two knowledge bases. They can be intuitively applied for an analysis of a degradation study that is introduced in the next section.

3 Phases of a Gray Box Degradation Study

The robustness of a knowledge system is investigated by a *degradation study*. Such a study considers one particular aspect of the robustness, e.g., its behavior with respect to noise in the input values or its behavior for biased knowledge. A degradation study consists of a collection of *degradation sequences*. Each degradation sequence is executed with the same settings and the result of the degradation study is determined by the averaged results of the degradation sequences. Every degradation sequence consists of a sequence of torture tests. In such a sequence the settings of the torture tests are subsequently changed, e.g., the input quality is gradually decreased. In every torture test all (possibly modified) cases are passed to the (possibly modified) system, and the accuracy of the system is measured. The results of the single torture tests are then used for computing the mean accuracy of the system in the particular degradation study. The elements of a degradation study are depicted in Figure 1.

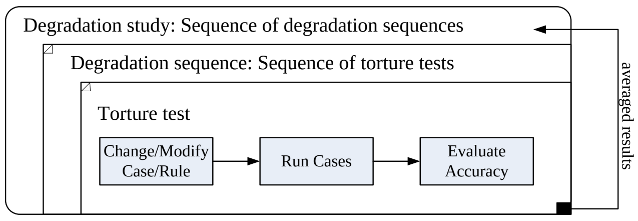
Degradation study: Sequence of degradation sequences
Degradation sequence: Sequence of torture tests
Torture test
Change/Modify Case/Rule
Run Cases
Evaluate Accuracy
averaged results

Fig. 1: Overview of the elements of a degradation study.

Before discussing the various types of torture tests we first consider the pre-analysis of the used data, i.e., the case base and the rule base.

3.1 The Pre-Analysis: Case Base and Rules

The analysis of the properties of the case base and the rule base is essential for a sound implementation of a degradation study. At best, all dimensions of the input-output space are uniformly distributed when investigating the used case base. A biased occurrence of parts of the input-output space would result in unbalanced results of the torture tests; then, some possible input-output combinations are missing and are consequently not investigated.

When concerning the rule base the perfect characteristics are not easy to define: in the best case the possible input space is not restricted by previously answered questions, i.e., the values of some inputs are only asked if values of previous inputs were assigned by the user. A typical example for such a restricted knowledge base is the implementation of a decision tree: depending on given input values other inputs are presented to the user to be answered. With the answers of the user the decision tree is traversed through a path until a leaf is reached which usually contains a solution. It is easy to see that such an interactive structure yields bad characteristics with respect to the robustness of the system, i.e., a falsely answered input will turn the dialog to another path of the tree and consequently will derive another or mostly no solution.

In the following we define measures for formally characterizing the properties of a case base and a rule base, respectively.

Case Base Properties When investigating the case base we consider the number of cases for each output, i.e., possible solution, and the number of findings typically contained in a case. A sound degradation study would work with a case base, that on the one hand contains equally distributed outputs, and on the other hand consists of equally sized cases (concerning the finding set).

Average Number of Cases (NOC) In a first step, the used cases can be characterized by the *average number of cases* for each output $o \in \Omega_{sol}$, i.e., the mean value with standard deviation of cases c with $o \in SOL_c$. A low number of cases makes it difficult to generalize the obtained results of the torture tests since only a small spectrum of the real world is possibly covered. A high deviation may indicate that some outputs only contain very few cases or a high number of cases. Consequently, the robustness can be very low or high for some outputs.

Average Number of Findings (NOF) A second analysis should consider the *average number of findings* contained in the cases, i.e., the mean value with standard deviation of the number of findings OBS_c for the cases c is computed. A low expectation value of findings can imply that the case base is not suitable for an extensive degradation study, because then large percentages of noise are required in order to actually modify the case; e.g., a noise level above 20% is needed to change at least one input value for a case base with a mean of 5 inputs per case.

Rule Base Properties A convincing degradation study should be also preceded by a careful investigation of the properties of the rule base. The number of rules for every output, the complexity of the rules, and the usage of the different types of rules are important indicators for the applicability of a degradation study.

Average Number of Rules (NOR) For degradation studies concerning the modification of the rule base the *average number of rules* for each output, i.e., the mean value with standard deviation of derivation rules for each output, is an interesting measure. Derivation rules for an output $o \in \Omega_{sol}$ are rules $r \in \mathcal{R}$ having output o in the rule consequent a_r. A low number of average rules or high deviation values can indicate that the results of the robustness studies may not be representative for all outputs contained in the rule base. The coverage of test cases has been investigated more thoroughly e.g. by Barr [6]. The analysis of the test cases is important for evaluating the sound execution of the degradation studies.

Types of Rules In classical systems the rule base only contained rules directly deriving an output for a given condition. However, for real-world applications the rule base can contain more refined types of rules. We distinguish three basic types of rules (cf. [7] for a more formal description):

1. *Derivation rules*: For a given condition the rule $r \in \mathcal{R}$ derives a specified output $o \in \Omega_{sol}$, i.e., having o in the rule consequent a_r. For example, with $a \in \Omega_a$, $v \in dom(a)$ and $o \in \Omega_{sol}$, and the rule

$$a : v \rightarrow derive(o)\,,$$

a solution o is derived for a given finding $a : v$. In detail, we distinguish derivation rules that derive an output categorically (as exemplified above) and derivation rules deriving an output using evidential categories, e.g., weighting scores or probabilities.

2. *Abstraction rules*: For a given condition such rules $r \in \mathcal{R}$ derive the value $v \in dom(a)$ for an intermediate abstraction $a \in \Omega_a$, that in turn can be also used in further rule conditions. For example, with $a_j \in \Omega_a$ and $v_j \in dom(a_j)$, and the rule

$$a_1 : v_1 \wedge a_2 : v_2 \rightarrow set\ value(a_3 : v_3)\,,$$

   the finding $a_3 : v_3$ is derived, if the two findings $a_1 : v_1$ and $a_2 : v_2$ are contained in the problem description. Abstraction rules are suitable for improving the reuse of existing knowledge or to enhance the design/understandability of a knowledge base.
3. *Indication rules*: These rules are used to implement an adaptive and efficient dialog of the system with a user. For a given rule condition the rule indicates new questions/inputs $a \in \Omega_a$ to be presented to the user, e.g., for $a_j \in \Omega_a$ and $v_j \in dom(a_j)$, and the rule

$$a_1 : v_1 \wedge a_2 : v_2 \rightarrow indicate(a_3)\,,$$

   the input a_3 is presented as a question to the user, when $a_1 : v_1$ and $a_2 : v_2$ holds. Such rules are commonly used for the implementation of a decision tree structure.

We see that a rule base containing not only derivation rules but also abstraction and indication rules is much more difficult to test for robustness. Thus, eliminating a specific question can prevent the system to ask the original questions of the given case, and thus entirely changes the semantics of the case. The availability of abstraction rules introduces rule chains in the knowledge base and therefore the elimination or modification of a specific input value can also change large parts of the original case. For this reason, no accurate evaluation may be possible.

Complexity of Rules For evaluating the results of degradation studies the complexity of the included rules is an interesting measure. In general, the complexity of a rule is calculated by the number of simple conditions (i.e., evaluating the value of a single input) included in the rule condition. This measure can be integrated in the previously described measure *Average Number of Rules (NOR)* by weighting the single rules with their complexity. Several of complexity measures focusing on the complexity of scoring rules were presented in [8].

3.2 Types of Torture Tests

As described earlier a degradation study considers the application of torture tests within a degradation sequence. We distinguish four different types of torture tests. It is important to notice that for one degradation study only one type of torture test is used. In the following we sketch the idea of the particular torture test types and motivate their applicability.

Torture by Case Input Deletion For every sequence in a degradation study a number of torture tests is executed: for each subsequent torture test a higher number of randomly selected inputs is not given as an input to the (unchanged) rule system. The test investigates the behavior of the system concerning an incomplete data acquisition, i.e., missing values.

Torture by Case Input Modification This test is similar to the case input deletion test above, but does not omit an increasing number of inputs from the cases. In contrast an increasing number of input values is modified before passing them to the knowledge system. With this type of torture test the robustness of the system with respect to an incorrect data acquisition can be evaluated, i.e., noise in the input space.

Torture by Rule Deletion The torture tests executed in degradation sequence are running the (unmodified) test cases with an decreased rule base quality: here, every subsequent torture test omits an increasing number of randomly selected rules during the problem solving session. The test can be useful to evaluate the robustness of the knowledge base with respect to an incomplete knowledge acquisition.

Torture by Rule Modification In contrast to omitting an increasing number of rules, as for the torture by rule deletion test, an increasing number of rules is *modified*. With this test the robustness of the knowledge system with respect to a biased or a faulty knowledge acquisition can be evaluated. For example, how dramatically do some incorrect rules worsen the accuracy of the system? In the context of our work we changed the weighting of the rule actions, i.e., the weighting of outputs, in order to simulate a biased domain specialist.

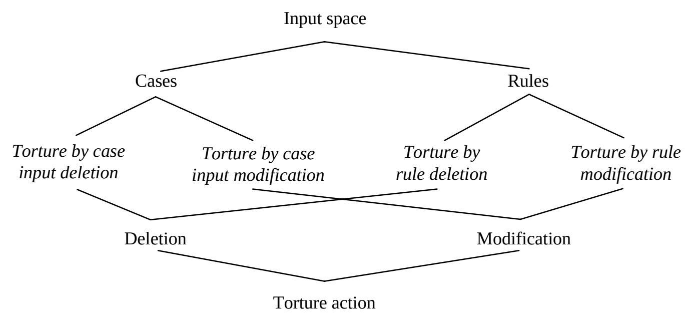
Input space
Cases
Rules
Torture by case input deletion
Torture by case input modification
Torture by rule deletion
Torture by rule modification
Deletion
Modification
Torture action

Fig. 2: A two-fold categorization of torture tests.

The four types of torture tests can be categorized in two ways: first, the tests can be classified according to the type of input that is target of the torture, i.e., tests concerning a worsening of the case base and the knowledge base, respectively.

Second, we can classify the tests according to the actual torture action that is performed during the study, i.e., there are tests fully deleting parts of the case base or rule base, and there are tests modifying parts of the case base or knowledge base. The two possible categorizations are depicted in Figure 2; the particular torture tests are written in italics.

3.3 Gray Box Testing with Background Knowledge

In a previous section we have motivated that the analysis of the case base and knowledge base is an essential precondition for robustness testing. The identified characteristics of the knowledge base are useful for predicting general robustness properties of the knowledge. Then, a rule base containing many indication and abstraction rules is more likely to be less robust than a knowledge base without a detailed dialog structure. Additionally, the complexity and type of rule conditions obviously plays a major role for the robustness properties: derivation rules with mostly single conditions should be indicators for a robust inference layer [1].

Furthermore, in a degradation study artificial noise is generated by randomly selecting the elements, i.e., input values or rule parts, for their deletion or modification. However, it is easy to see that in a real world setting not all input values have an equal "probability" to be falsely answered by the user. For example, inputs only relevant for the dialog structure are commonly answered correctly, since deciding about answers for such questions is mostly very easy. In the following we present two disjoint types of background knowledge in order to conduct more realistic degradation studies even for knowledge bases with a dialog structure.

Since the actual implementation of the knowledge base is considered and background knowledge is used for guiding the torture tests we call such a degradation study a *gray box test*.

Preserving Important Inputs As discussed above some input values are very unlikely to be answered incorrectly by the user but can be very important for the dialog structure. In a gray box degradation study we can provide a set of such important inputs.

Then, the *important inputs* $\mathcal{IV}_a \subset \Omega_a$ are categorically excluded from the elimination and modification procedures of the torture tests. The set of important inputs is disjoint with the set of ambivalent inputs that is introduced in the following.

Ambivalence of Inputs In contrast to the definition of important inputs the procedure of a degradation study can be also directed by the definition of *ambivalent inputs*. In a real world setting there often exists a set of inputs that are more likely to be mixed up by a user than other inputs. Such inputs commonly have a large range of possible values or the actual answer is difficult to give. For example, in the medical domain some inputs like the age or height of a patient are very simple to obtain, whereas other inputs like the findings of a liver examination are more difficult to acquire.

A set of ambivalent inputs $\mathcal{AI}_a \subset \Omega_a$ can be defined for a degradation study, where $\mathcal{AI}_a \cap \mathcal{IV}_a = \emptyset$. In consequence, these inputs are selected with a higher probability

[1] However, it is clear that the formalization of domain knowledge often requires the conjunction of many single conditions.

during the modification/deletion procedures of the torture tests. A typical setting for a degradation study would be that inputs $i \in \mathcal{A}I_a$ are selected with a doubled probability for torturing than inputs $i' \in \Omega_a \setminus (\mathcal{A}I_a \cup \mathcal{I}V_a)$.

The presented types of background knowledge only cover the tests modifying or deleting the inputs from cases. Here, for a case c the number of questions to be modified or deleted is not randomly selected from the entire set OBS_c but from a restricted input space, e.g., $OBS_c \setminus \mathcal{I}V_a$ for a degradation study preserving important inputs. The second type of background knowledge even concerns the randomness of the selection. Up to now we have not considered background knowledge that adapts torture tests concerning the deletion or modification of the rule base.

4 Case Study

We demonstrate the presented work by two case studies, one testing the robustness of a biological consultation system, and one testing the robustness of a medical consultation system. The two corresponding rule bases show different characteristics with respect to their implementation structure: Whereas, the biological rule base mainly consists of simple rules and no complex dialog control, the medical rule base contains many indication rules for the implementation of a sophisticated dialog structure. Furthermore, this rule base has a large portion of complex derivation rules.

4.1 Degradation of a Plant Rule Base

The first case study was conducted in the biological domain with the rule base of a plant consultation system [9]. This system identifies the most common flowering plants vegetating in Franconia, Germany. For a classification of a given plant the user enters findings concerning the flowers, leaves and stalks of the particular plant. Since the system was planned to be used by non-expert users the *diagnostic scores* [10] pattern was applied. This knowledge formalization pattern proposes to implement only derivation rules with single conditions, i.e., conditions over one finding, in order to increase the robustness of the system, e.g., concerning possibly erroneous data acquisition.

The rule base contains 5623 derivation rules, and the used case base consists of 96 cases (with a mean of 40 different inputs and in total 39 distinct outputs). All cases are correctly solved with the original version of the rule base.

For each degradation study 5 iterations of degradation sequences were performed with 20 degrading torture tests in each sequence (plus one initial torture tests with 0% modification, i.e. 100% quality). Since all cases contained only single outputs as solution for a case we used the measures precision and recall for the evaluation of the accuracy. The precision calculates the degree of inferred outputs that are actually correct, and can be defined as follows:

$$precision(SOL_c, SOL_{c'}) = \begin{cases} \frac{|SOL_c \cap SOL_{c'}|}{|SOL_{c'}|} & \text{if } SOL_{c'} \neq \{\}\,, \\ 1 & \text{if } SOL_{c'} = \{\} \text{ and } SOL_c = \{\}\,, \\ 0 & \text{otherwise.} \end{cases}$$

where $SOL_c, SOL_{c'}$ are the output of the stored case c and the derived output of the problem solving case c', respectively. The *recall* measures the degree of expected outputs that are actually derived, and is defined by

$$recall(SOL_c, SOL_{c'}) = \begin{cases} \frac{|SOL_c \cap SOL_{c'}|}{|SOL_c|} & \text{if } SOL_c \neq \emptyset \\ 1 & \text{otherwise} \end{cases},$$

where $SOL_c, SOL_{c'}$ are the output of the stored case c and the derived output of the problem solving case c', respectively.

Case Torture Tests The first degradation study used the torture test type *case input deletion* in order to see how the system behaves with respect to missing values. The second degradation study used the torture test type *case input modification* simulating incorrect user inputs. In Figure 3 the precision values of the two degradation studies

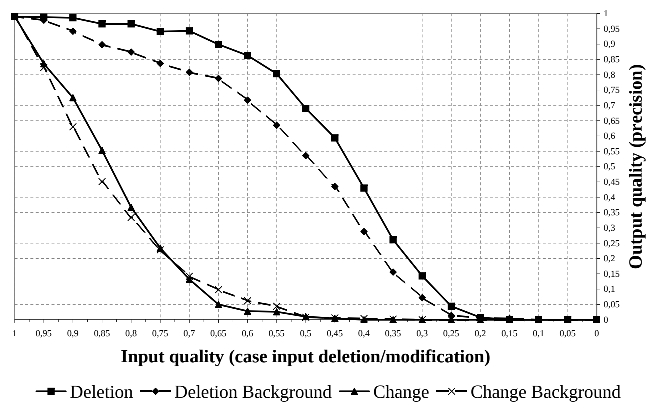

Fig. 3: Precision values for the plant system without and with ambivalence knowledge (dotted lines): precision values of the case input deletion tests are displayed as lines with squares and rhombus, respectively. Precision values of the case input change test are displayed as triangled and crossed lines, respectively. Degraded input quality is displayed on the x-axis.

are displayed: on the x-axis the degrading input quality (less user inputs vs. less correct user inputs) are displayed; on the y-axis the averaged precisions of the particular torture tests are depicted. Precision values of the case input deletion tests are displayed as lines with squares and rhombus, respectively. Precision values of the case input change test are displayed as triangled and crossed lines, respectively. The tests uncovered that for an incomplete data acquisition the system has an acceptable precision for an input quality between 70% and and 100%, i.e., the system is able to take about 30% missing data. In contrast, the system lacks an acceptable precision for faulty data input, since the precision falls below 85% for an input quality less than 95%. In Figure 4 the recall values of the degradation studies are shown. Recall values of the case input deletion tests

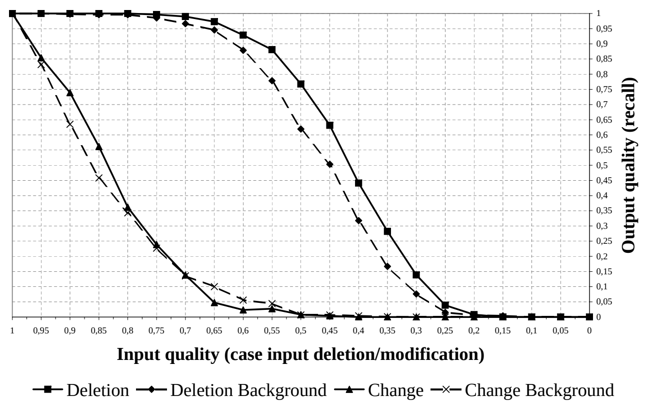

Fig. 4: Recall values for the plant system without and with ambivalence knowledge (dotted lines): recall values of the case input deletion tests are displayed as lines with squares and rhombus, respectively. Recall values of the case input change test are displayed as triangled and crossed lines, respectively. Degraded input quality is displayed on the x-axis.

are displayed as lines with squares and rhombus, respectively. Recall values of the case input change test are displayed as triangled and crossed lines, respectively. Degraded input quality is displayed on the x-axis. Here, we see a similar behavior as described for the precision of the system: The expected solutions of the cases were also derived by the system for a completeness level greater than 65%. However, the system is very sensitive for changed input values and is only able to derive the expected solutions for a completeness level greater than 95%. Concerning the application of background knowledge we see that the increased omission of ambivalent inputs yield a worse accuracy both for the precision and recall of the plant system. Surprisingly, the (slight) modification of input values of ambivalent inputs had no significant effect on the reasoning accuracy of the system. This behavior can be explained by the fact that there exist similar rules for ambivalent inputs in the rule base. A change of such values therefore resulted in the activation of rules with similar or equivalent consequents. In general, the degradation studies with respect to the case input showed a monotonic derivation quality (see Definition 5).

Rule Torture Tests Figure 5 displays the results of the torture tests deleting and modifying parts of the rule base. Here, the precision as well as the recall values of the tests are shown. In the upper part of the figure we see the precision and recall values for the torture test *rule modification*. The rule base consists of rules $a : v \rightarrow derive(o)$, where $derive(o)$ either adds a positive or negative weight to the given output o. In this test the weights were randomly reduced or increased by one weight category when selected for the torture test. Interestingly, the modification of the rule weights did not affect the precision and the recall significantly. Moreover we see that the tests uncovered a non-monotonic derivation quality, i.e., a worse input quality can result in a better derivation

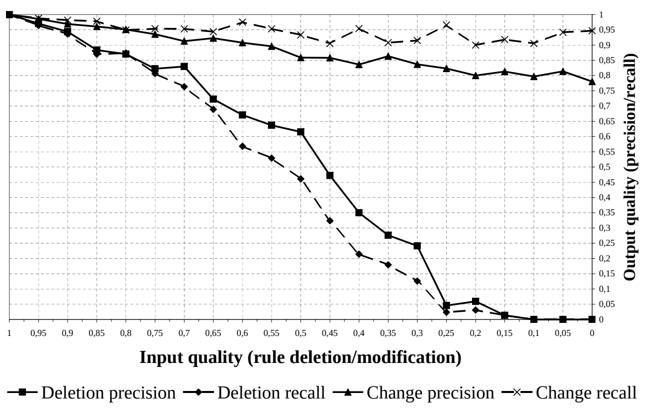

Fig. 5: Precision and recall values for the plant system while degrading the quality of the rule base: precision values of the rule deletion tests are displayed as lines with squares; the recall for the deletion test is depicted with a dotted line and a rhombus. Precision values of the rule change test are displayed as triangled lines; the recall of this test is displayed with crossed dotted lines. Degraded input quality is displayed on the x-axis.

quality. The lower part of the figure depicts the precision and recall values for the torture test *rule deletion*. Here we see that an increased elimination of rules result in an almost monotonic decrease of the output quality. The improvements of the derivation quality can be explained by the stochastic variance of the tests: for each torture test new rules are randomly selected and removed. Here one collection of selected rules had more impact on the reasoning behavior than another collection of rules.

4.2 Degradation of the SonoConsult Rule Base

The second case study considered the degradation of the medical knowledge system SonoConsult [11], a consultation system for sonographic examinations. Here, 427 inputs and 221 outputs are structured in a taxonomy, where only a small portion of "leaf" inputs and outputs are actually visible to the user. The rule base consists of 4234 derivation rules, 2298 abstraction rules, and 2151 indication rules implementing a complex dialog structure. Due to its different characteristics the system was not supposed to be as robust as the previously described plant system. The torture tests were performed using 250 cases, where each case contains about 60 findings as the problem description and a mean of 6 outputs as its solution. Due to the multiple faults (outputs) characteristics of the case base the application of the measures precision and recall seemed to be not appropriate; the E-measure combining both, precision and recall, was used instead. The general E-measure is defined as follows:

$$E(c, c') = \frac{(\beta^2 + 1) \cdot \mathit{precision}(SOL_c, SOL_{c'}) \cdot \mathit{recall}(SOL_c, SOL_{c'})}{\beta^2 \cdot \mathit{precision}(SOL_c, SOL_{c'}) + \mathit{recall}(SOL_c, SOL_{c'})} ,$$

where $SOL_c, SOL_{c'}$ are the output of the stored case c and the derived output of the problem solving case c', respectively. In the case study we used $\beta = 1$ which is balancing the precision and the recall, i.e., the F-measure.

Case Torture Tests Figure 6 shows the F-measure values of the degradation studies for the torture test type *case input deletion* and the type *case input change*. In the case study 5 degradation sequences were performed with a quality stepwidth of 10% resulting in 10 torture tests for each degradation sequence (plus one torture test with 100% input quality). On the x-axis the degrading input quality (less user inputs vs. less

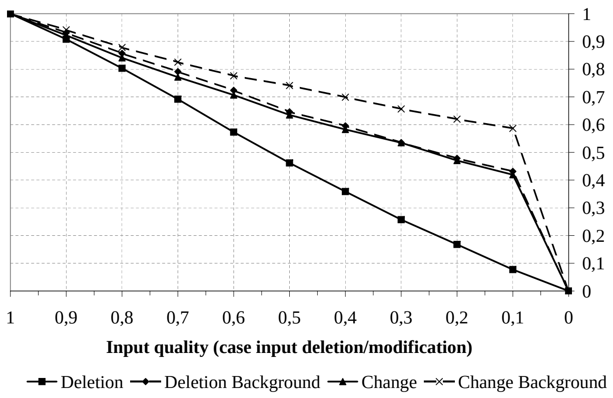

Fig. 6: F-measure values without and with important inputs knowledge (dotted lines): F-measure values of the case input deletion tests are displayed as lines with squares and rhombus, respectively. F-measure values of the case input change test are displayed as triangled and crossed lines, respectively. Degraded input quality is displayed on the x-axis.

correct user inputs) are displayed; on the y-axis the averaged F-measure values of the particular torture tests are depicted. F-measure values of the case input deletion tests are displayed as lines with squares and rhombus, respectively. F-measure values of the case input change test are displayed as triangled and crossed lines, respectively. The results confirm the expectations that the rule base only provides a limited robustness. The inclusion of background knowledge weakens the trend only for lower quality values, which can be explained that even more important inputs should be included as background knowledge; up to now only 35 inputs are marked as important and some more indication questions are possibly contained in the rule base. The similar behavior for the delete inputs tests and the change input value tests suggests, that even slightly changing an input value has the same worsening behavior like completely omitting the input.

Rule Torture Tests Figure 7 shows the F-measure values of the degradation studies of the torture test types *rule deletion* and *rule change*. In the case study 5 degradation sequences were performed with a quality stepwidth of 5% resulting in 20 torture tests for each degradation sequence (plus one torture test with 100% input quality). The

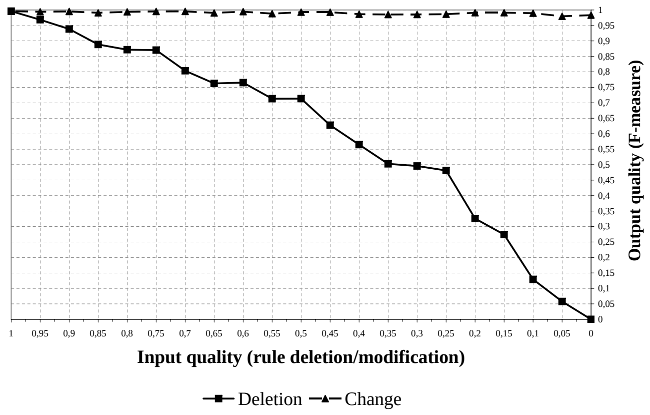

Fig. 7: F-measure values of the rule deletion test are displayed as lines with squares; F-measure values of the rule modification test are displayed as triangled and dotted lines. Degraded input quality is displayed on the x-axis.

results for the torture test *rule change* were quite surprising since slight changes of the rule weights did not yield a significant decrease of the derivation quality. As for the plant system the rule base also contained redundant knowledge that compensates possible changes of the rule weights. In contrast, the deletion of rules yielded an almost monotonic decrease of the output quality. The slight improvements of the derivation quality for input qualities 0.6 and 0.5 can be explained by the stochastic variance of the tests as already described in the case study with the plant system. We expect that for a larger number of degradation sequences this variation will be removed and the results will show a strict monotonic behavior. In the final version of this paper we will present a degradation study with more degradation sequences; currently one degradation study with 5 sequences takes about 8.5 hours running on a standard computer.

5 Conclusion and Outlook

Rules are an intuitive and simple representation for the formalization of domain knowledge. In the past, there has been much research on general validation and verification methods for rule bases. However, for the practical applicability of intelligent systems also their robustness is of importance. In this paper, we presented an approach for testing the robustness of rule bases extending previous approaches (cf. [5]) by background knowledge and an intensive pre-analysis, thus yielding a gray box test. It is worth noticing that on the one hand the pre-analysis of the knowledge base and the case base is necessary for a sound interpretation of the study results. On the other hand, background knowledge is used for a more concise definition of the robustness tests. The work was demonstrated by a case study reporting robustness tests on two larger rule bases taken from the biological and the medical domain. As a general contribution, degradation studies can help to determine the self-confidence level of a system: for example, if a given number of omitted inputs is exceeded, then the system can retain from giving a solution, thus avoiding to give a wrong solution with a high probability.

In the future, we are planning to improve the applicability of background knowledge: first, currently no background knowledge for the adaptation of rule torture tests is considered. For example, when using rules deriving an output with a certain probability we can expect that rules with a medial probability are more likely to be biased or noisy than rules with probabilities near to 1 or 0. Second, the definition of background knowledge can be simplified by automatically generating proposals for important inputs and ambivalent inputs. Important inputs typically are inputs contained in indication rules but not in derivation or abstraction rules. An input can be considered as an ambivalent input if the corresponding input values are occurring very frequently in rules for different outputs.

References

1. Ayel, M., Laurent, J.P.: Validation, Verification and Test of Knowledge-Based Systems. Wiley (1991)
2. Preece, A., Shinghal, R.: Foundation and Application of Knowledge Base Verification. International Journal of Intelligent Systems **9** (1994) 683–702
3. Knauf, R.: Validating Rule-Based Systems: A Complete Methodology. Shaker, Aachen, Germany (2000)
4. Horrocks, I., Patel-Schneider, P.F., Bechhofer, S., Tsarkov, D.: OWL Rules: A Proposal and Prototype Implementation. Journal of Web Semantics **3**(1) (2005) 23–40
5. Groot, P., van Harmelen, F., ten Teije, A.: Torture Tests: A Quantitative Analysis for the Robustness of Knowledge-Based Systems. In: Knowledge Acquisition, Modeling and Management. LNAI 1319, Berlin, Springer (2000) 403–418
6. Barr, V.: Applications of Rule-Base Coverage Measures to Expert System Evaluation. Knowledge-Based Systems **12** (1999) 27–35
7. Baumeister, J.: Agile Development of Diagnostic Knowledge Systems. AKA Verlag, DISKI 284 (2004)
8. Atzmueller, M., Baumeister, J., Puppe, F.: Semi-Automatic Learning of Simple Diagnostic Scores utilizing Complexity Measures. AI in Medicine **37**(1) (2006) 19–30
9. Ernst, R.: Untersuchung verschiedener Problemlösungsmethoden in einem Experten- und Tutorsystem zur makroskopischen Bestimmung krautiger Blütenpflanzen [Analysis of various problem solving methods with an expert and tutoring system for the macroscopic classification of flowers]. Master's thesis, University Würzburg, Biology department (1996)
10. Puppe, F.: Knowledge Formalization Patterns. In: Proceedings of PKAW 2000, Sydney, Australia (2000)
11. Hüttig, M., Buscher, G., Menzel, T., Scheppach, W., Puppe, F., Buscher, H.P.: A Diagnostic Expert System for Structured Reports, Quality Assessment, and Training of Residents in Sonography. Medizinische Klinik **3** (2004) 117–22