Ethics and Information Technology (2022) 24:19
https://doi.org/10.1007/s10676-022-09645-y

ORIGINAL PAPER

Reflection machines: increasing meaningful human control over Decision Support Systems

N. A. J. Cornelissen[1] · R. J. M. van Eerdt[1] · H. K. Schraffenberger[1] · W. F. G. Haselager[2]

Accepted: 23 February 2022 / Published online: 21 March 2022

Abstract
Rapid developments in Artificial Intelligence are leading to an increasing human reliance on machine decision making. Even in collaborative efforts with Decision Support Systems (DSSs), where a human expert is expected to make the final decisions, it can be hard to keep the expert actively involved throughout the decision process. DSSs suggest their own solutions and thus invite passive decision making. To keep humans actively 'on' the decision-making loop and counter overreliance on machines, we propose a 'reflection machine' (RM). This system asks users questions about their decision strategy and thereby prompts them to evaluate their own decisions critically. We discuss what forms RMs can take and present a proof-of-concept implementation of a RM that can produce feedback on users' decisions in the medical and law domains. We show that the prototype requires very little domain knowledge to create reasonably intelligent critiquing questions. With this prototype, we demonstrate the technical feasibility to develop RMs and hope to pave the way for future research into their effectiveness and value.

Keywords AI ethics · Meaningful human control · Responsibility gap · Human-machine interaction · Decision Support Systems

Introduction

Increasingly, aspects of decision making are being taken over by Artificial Intelligence. In many domains, including finance, law, healthcare, insurance, dating and more, algorithms have already been introduced. These algorithms inform, prepare, or generate hypotheses, diagnoses, triage, and choices. In many, if not all, of these domains, principled questions have been raised about the extent to which humans could and should remain in control over the overall process.

✉ W. F. G. Haselager
pim.haselager@donders.ru.nl

N. A. J. Cornelissen
niels.cornelissen@ru.nl

R. J. M. van Eerdt
ralph.vaneerdt@ru.nl

H. K. Schraffenberger
h.schraffenberger@ru.nl

[1] Radboud University Nijmegen, Nijmegen, The Netherlands

[2] Donders Institute for Brain, Cognition and Behaviour, Radboud University Nijmegen, Nijmegen, The Netherlands

For example, article 22, paragraph 1 of the General Data Protection Regulation of the European Union (https://gdpr.eu/article-22-automated-individual-decision-making/) states that a "data subject shall have the right not to be subject to a decision based solely on automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly affects him or her." Although the precise limitations and implications of article 22 have been subject to discussion (Roig, 2017; Wachter et al., 2017), paragraph 3 specifies that this implies in some cases "the application of suitable measures to safeguard the data subject's rights and freedoms and legitimate interests, at least the right to obtain human intervention".

Article 22 has played an important role in shaping the interaction between humans and intelligent machines by reinforcing having humans 'on' the information processing loop. Instead of being continuously fully immersed 'in' the loop, by taking the decisive steps from input to output, humans could take on a monitoring role, supervising machine information processing and intervening or overriding it when necessary. Practically speaking, this has further stimulated the development and application of so-called Decision Support Systems (DSSs). These systems do not

independently take decisions but instead, suggest or recommend hypotheses or decisions to human supervisors who approve or reject them. These human users of DSSs monitor the operations or at least the output of the AI and can (and should) intervene when necessary. However, the question is whether humans can consistently fulfil such a supervisory role over prolonged periods of time. One particular concern is that humans are held responsible for decisions that, when considering the human cognitive abilities to actively monitor DSSs, in reality, are out of their control. For instance, humans might not be capable of exercising the degree of attention that their being 'on' the loop requires for general psychological reasons (Eysenck & Keane, 2002). Hence, they might over time become unable to fulfil their monitoring role meaningfully, but rely on the DSS, providing mere human stamps of approval. Indeed, there is a considerable risk that humans become overly reliant on DSSs (Grissinger, 2019; Liu, 2018). Various factors can play a role here, tiredness, recklessness, boredom, etc. These consequences have been discussed under a variety of labels, most prominent amongst which are 'meaningful human control' (Mecacci & de Santoni, 2020; de Santoni & van den Hoven, 2018), 'automation complacency' (Merritt et al., 2019) and 'responsibility gaps' (Matthias, 2004; de Santoni & Mecacci, 2021).

In the light of these risks of DSSs use, this paper explores an additional support system for humans on the loop, which we call the 'Reflection Machine' (RM). In general, RMs provide feedback on human decisions, intending to increase the control and involvement of humans in the decision-making process. In the context of a DSS, an RM aspires to improve human supervision over DSSs by asking questions about the reasons for accepting or rejecting a recommendation of a DSS. Simply put, whereas a DSS thinks 'for' the human, the RM thinks 'against' them. It can do so by pointing at data that could be inconsistent with an accepted decision or suggesting an alternative hypothesis based on some case-specific information.

The envisioned concept of an RM can be illustrated with an example of how the system ideally could perform in the future. Imagine a parent and child consulting their general practitioner because the child has been suffering from abdominal pain. Such a symptom can be caused by a wide variety of physiological issues (such as appendicitis or first teeth coming through), but also by a plethora of social factors (e.g., stress, anxiety, or domestic abuse). The practitioner performs the standard procedure for such a case (including physical tests, asking for other symptoms, etc.) and enters the resulting information into the computer. The GP uses decision support software that (based on the entered information) suggests the abdominal pain is likely a symptom of a winter flu. Without an RM, the practitioner might be inclined to agree with this reasonable conclusion without giving it much thought. In contrast, when using an RM in conjunction with the decision support software, the RM could identify factors that the GP might have overlooked. This would encourage the GP to question the decision by pointing out additional factors that could be considered and emphasising information that points towards other conclusions. For instance, a future RM could consider that the GP may not have taken the child's social situation sufficiently into account. It thus could ask a question like "did you consider how the child experienced school this week?". More generally put, example questions from an RM to the human supervisor could take the form: "Did you consider factor x?" and "Which role did data point y play in reaching your conclusion?". The general underlying idea is that while thinking 'for' the human could lead to more superficial involvement of humans in decision making, thinking 'against' might deepen their involvement.

Of course, thinking 'against' comes at a price, effectively asking more time and effort of a human user. Hence the balance between the activities of the DSS and RM will be an important topic for further research. At this point, our aim is to demonstrate the in-principle technical feasibility of an RM by developing and exploring a first prototype. The main body of this article is dedicated to demonstrating how RMs can be implemented and it is organized as follows: First, we briefly discuss design choices in the implementation of RMs and then move on to a proof-of-concept implementation that focuses on decision making in the domains of medicine and law: We indicate how we simplified the task of decision making in these domains to allow for rapid prototyping. We then describe the considered problem cases in sufficient detail to make the case related aspects of the implementation understandable. Subsequently, the prototype RM and its basic implementation details are presented. Finally, we present ideas for future work on RMs and suggest how to investigate their effectiveness empirically.

The design space of RMs

DSSs typically aim at helping users make 'good' decisions and support decision-making by providing information and suggestions *before* the human decision is made. In contrast, RMs *respond* to a human decision, with the goal to increase the human's involvement, agency, and control in the decision-making process, which could lead to a change in the decision. While all RMs have this in common, they still can take many different forms. Therefore, when implementing an RM, some general design choices need to be made, ideally by considering the intended use case and the needs of prospective users. This article describes the proof-of-concept implementation of an RM for the medical and law domains.

When designing an RM, one crucial consideration is the *decision-making flow* in which the system will be used. On

the one hand, an RM can supplement independent human decision making. On the other hand, an RM can be used in the context of DSSs. This is the focus of this article, as a central goal of RMs is to reduce overreliance on DSS and passive decision-making when using a DSS, aiming to prevent issues such as the responsibility gap. When used alongside a DSS, the DSS suggests a solution to a given problem. The user can then make a decision, either following the recommendation of the DSS or deviating from it. After this step, the RM presents its feedback and allows the user to change their decision. This is also the envisioned flow in our prototype. For instance, we have implemented two health care scenarios (see below) where the user is presented with background information about a patient and their symptoms, gets a suggestion about the underlying illness from the DSS, and is then asked to make a diagnosis. Once the user decides, they are presented with a response from the RM. Ultimately, they can either stick with or change their diagnosis.

Another consideration is *how to present the feedback* to the user. In most use cases, a short text (presented visually or via audio) will be adequate. Our proof of concept presents text-based visual feedback. A follow-up question is what form this text takes. Two main possibilities include *statements* and *questions*. As described in the example with the GP, the RM could ask whether certain factors (e.g., school situation, home situation) were considered in the diagnoses. Similarly, a statement could, for instance, point out that the child's recent change in eating habits suggests another diagnosis. We expect questions to engage users more and thus have opted for a question-based design in our prototype.

The actual questions or statements can also take many forms. In particular, the feedback can vary with respect to *how strongly it relates to the decision at hand* and *how 'intelligent' it is*. In a most basic form, an RM could ask the question "Are you sure?" independent of the specific situation. In interaction design, such forms of friction are used to prevent mistakes (such as deleting important files) and have been suggested to prevent mindless interactions and prompt reflection (Cox et al., 2016). Our prototype also uses the question "Are you sure?" as a minimal response. However, we expect RMs to be more effective if the feedback relates to the user's actual choice or behaviour. Hence, the prototype also provides more intelligent feedback that emphasizes factors that likely were overlooked or misinterpreted. These questions currently take the form "Did you consider the following?" followed by such a factor. Unlike a DSS, the RM thus does not suggest a concrete solution - it is up to the user to draw their own conclusions from the presented feedback. Possible approaches to arrive at RM questions and criteria to evaluate them are one main focus of this article and discussed in detail in the "Implementation methods" section.

Finally, the *triggering conditions* and *frequency* of the RMs machine's response need to be determined. In an extreme case, an RM could respond to every human decision with a series of questions. However, repeatedly presenting similar questions to the user could cause habituation and decrease their response to the RM's questions over time (see, e.g., https://explorable.com/habituation; and Kieffer 2017). Thus, more refined triggering conditions should be considered. As illustrated in the envisioned scenario with the GP, the RM *ideally* responds when little human reflection went into a decision and when the decision of the DSS is accepted or rejected without giving this sufficient thought. One strategy to determine whether the RM should present a question thus could be to estimate whether the user has been actively involved in the decision making process (e.g., by taking into account whether the DSS solution has been accepted and considering how much time went into reaching a conclusion). Another strategy could assume that human involvement likely has not been sufficient if important factors seem to have been overlooked or misinterpreted, resulting in a solution that does not seem ideal. In this project, the presented RM prototype responds to every human decision, independently of whether the DSS feedback has been accepted or not. The exact response is determined by taking the RM's confidence in the user's solution into account. If the confidence is low, the RM asks about one of the factors that cause the low confidence in the decision. This way, we hope to point the users' attention to certain pieces of information that seem to have not received enough attention (either have been overlooked or misinterpreted).

Problem simplification

Responsibility gaps can occur in any domain where computer-assisted decision making is possible, and the number of affected domains grows as more DSSs and semi-autonomous agents are developed. DSSs can play an important role in solving complicated police tasks (Dechesne et al., 2019). Dual mode vehicles are a perfect example of (semi)-autonomous systems (Mecacci & Santoni de Sio, 2020). In both law and medical domains, computer-assisted decision making may increase the accuracy and efficiency of various processes, such as diagnosing patients (Kruse & Ehrbar, 2020). These applications apply to open-ended decision-making tasks where a ground truth is not always available. To solve such open-ended problems, big datasets may be required. These datasets might be available already depending on the domain or may, for example, be generated for systems like IBM Watson (Chen et al., 2016). For a rudimentary prototype, however, the requirement of big data makes it difficult to work with real-world cases.

Hence, to allow for a first exploration of a prototype, these open-ended problems were adapted into more suitable

multiple-choice tasks (van Eerdt, 2021). The resulting decision cases, two for the medical domain and two for the law domain, describe a medical or a law situation and a related question, four possible answers to that question and an answer as suggested by a DSS. While the ground truth to these problems remains unknown, much like the cause for the abdominal pain in our practical example, the decision case is simplified enough to be addressed by a basic RM with minimal domain knowledge. Thus, a prototype can be developed that can be tested on feasibility and effectiveness and eventually be scaled up to solve more open-ended problems.

Case description

As stated before, four decision cases were created to explore the feasibility of introducing an RM in DSS-supported decision-making processes. All decision cases present a multiple choice question with four possible answers. Two of the decision cases use a medical scenario and two take place in the law domain. The medical cases revolve around making a diagnosis based on symptoms and background information on the patient. The medical case studies and their solutions were created in collaboration with a fifth-year medical student to ensure the tasks are complex and challenging enough for a medical student to give a substantiated answer. For the patients' backgrounds, the medical case descriptions were constructed using a table of patient information options (*Van der Stigchel et al. submitted; Van der Stigchel*, 2021). Each case is built around one main symptom, which can be caused by multiple different illnesses, of which four are presented as possible answers. The symptoms and illnesses were collected from http://www.huisartsdd.nl/. For each illness, a possible RM question was added to 'counter-think', i.e. argue against the link between the illness and the main symptom. Finally, all this information combines to form a case presentation according to Budgell's guidelines for case studies (Budgell, 2008).

For the first medical case, the main symptom is *involuntary weight loss*, and the main symptom of the second medical case is *coughing*. The illnesses that could cause involuntary weight loss are colon cancer, depression, hyperthyroid, diabetes mellitus type 2. The illnesses that could cause coughing are lung cancer, upper respiratory tract infection, a side effect of the medicine lisinopril and gastroesophageal reflux disease. These illnesses will be the four possible answers to their corresponding cases. The cases are expanded with more symptoms connected to the illnesses and information from the different categories from the table from Van der Stigchel (2021). In the end, all the information is combined in a case presentation (Budgell, 2008).

To construct the law cases, two simplified versions of real past cases were reviewed. To adapt the original law cases into a suitable form, there was a collaboration with a third-year law student. This ensured that the cases used the appropriate law terminology and were complex enough to simulate a real law case.

The first case is based on the Bijlmer case (https://www.hetrechtenstudentje.nl/jurisprudentie/eclinlhr1984ac8567-bijlmer-noodweer-arrest/), which involved, among other, the question of whether the suspect in a shooting has acted in self-defence. For the second case, a simplified version of case number 09/900065-05 is used (http://deeplink.rechtspraak.nl/uitspraak?id=ECLI:NL:RBSGR:2005:AT8463). This case involves a group of suspects throwing a tile from a viaduct onto a passing car, causing the driver's death. Sections of the real law cases were excluded to simplify it, while still resembling a real law case. For both law cases, four possible rulings were created to fit the multiple-choice format. The judge's decision was used to inform one of the four multiple-choice solutions for each case.

Note that there is no *correct* answer to the four cases. All answers are plausible because features connected with each answer are all present in the case presentation. Due to the nature of these tasks however, especially for the law cases which have previous real-world verdicts, one or more solutions may emerge to be optimal or preferred, though this is not by design.

Implementation methods

Prototype specification

How should the RM respond to a user's decision? As mentioned, different approaches can be used to determine the feedback. To allow for experimentation with (a combination of) different approaches, a modular design is required. Robbins proposed a classification of critic modules that served purposes similar to that of an RM (Robbins, 1998). These modules were designed to analyse different parts of a problem and compare them to a submitted user solution. In line with Robbins' classification, multiple modules for an RM are described below. For the sake of readability, we focus on the two medical use cases. However, the RM design likewise applies to the law cases.

Modules 0a and 0b serve as a control group, using *unintelligent* methods to challenge the user's solution. Modules 1, 2 and 3 present *more intelligent and informed methods* that aim to produce more relevant and helpful feedback than that of the control group:

- 0a, The RM should at least produce "Are you sure?".

This is an uninformed control case, from here on referred to as the 'uninformed' module. In the abdominal pain example presented earlier, the RM would simply provide this question to the GP.

- 0b The RM should ask about aspects specific to the case.

  This is an informed control case, from here on referred to as the 'informed' module. Regarding the abdominal pain example, the informed module would for instance ask "Are the child's teeth breaking through?".
- 1 The RM should check for alarming discrepancies between the diagnosis of the DSS and the general practitioner.

  Assuming that the DSS is good enough to be used in the heavily regulated domain of healthcare, its solution should be as close as we can get to a ground truth. Thus, this module is akin to a correctness critic, from here on referred to as the 'correctness' module. The correctness module would for instance note that whereas the DSS based its decision on data points about "the school situation", the GP mainly decided on the basis of "eating habits", and ask a question related to this discrepancy.
- 2 The RM should check for alarming discrepancies between the diagnosis of the practitioner and the symptoms described by the patient.

  This module aims to verify that the diagnosis is consistent with the patient's symptoms. This is akin to a consistency critic, from here on referred to as the 'consistency' module. This module would for instance note that the GP's diagnosis of winter flu would not fit well with the absence of fever and ask a question about that.
- 3 The RM might produce a differential diagnosis of its own to compare with the user's solution.

  This module is closely related to an alternative critic, from here on referred to as the 'differential' module. The RM would come up with for instance the alternative of child abuse in contrast with the GPs focus on the child's diet, by asking e.g. "Was it checked whether the child has any bruises?".

Unlike Robbins' critic modules, where each module produces valuable critiques to the user's solution, the modules might not all produce useful insight. In addition, unlike a critic, the goal of the RM is not to improve the user's solution but rather to increase the user's involvement in the decision-making process while working with a DSS, thus increasing meaningful human control. Hence, to implement and evaluate a simple prototype and to avoid the risk of overloading the user with information, the output of the RM was limited to one counter question only. The diagram in Fig. 1 describes how the different modules can work alongside one another, while the RM only produces one output.[1]

Our RM (particularly correctness and consistency modules 2 and 3) needs to 'reason' about possible solutions to a given problem. For this, the design of an RM can learn from the design of DSSs. Classic DSSs use 3 main components to base their decision on. Phillips-Wren et al. (2009) identified these 3 main components as a knowledge base, a data base, and a model base. Since the model base, which classically contains formal models of decision making, is largely dependent on the content and representation of these formal models, this component will vary across implementations and problem domains.

For this simple prototype, the knowledge base and data base are combined in one table. This yields 4 inputs for the machine:

- The textual case information as presented to the human expert,
- A one-word textual representation of the solution as proposed by the DSS. (Though not strictly required for questioning the user's solution, it makes sense to include this input in the context of responsibility gaps.) For each of the four tasks, the recommendation by the DSS was determined during the design of the case studies, choosing an arbitrary (not ground truth) multiple-choice answer.
- A one-word textual representation of the solution as submitted by the human user (similarly, the RM can function without this input to question a DSS, though in the context of responsibility gaps that is not the goal),
- A lookup table that holds information on possible solutions and their related features.

Using these inputs, the prototype should produce feedback for the user. To this end, n modules map the input to intermediate output. Any module must satisfy the following constraint to function within the prototype, effectively mapping a subset of the above-defined input to a list of potential questions and a set of weights representing their value:

$$\left(\subseteq \left[case,\ support\ solution,\ user\ solution,\ lookup\ table\right]\right) \rightarrow \left[(question,\ keyword, confidence, multiplier),\ \ldots\right]$$

Where a complete entry in the list of RM questions is defined as a set of:

- A generated question, as it would be presented to the user,

[1] Created using Google Draw: https://docs.google.com/drawings/d/1rPQBUOy1_LdIfOFodENEMi8AT4Duv859GjFzI2YBTw0/edit?usp=sharing

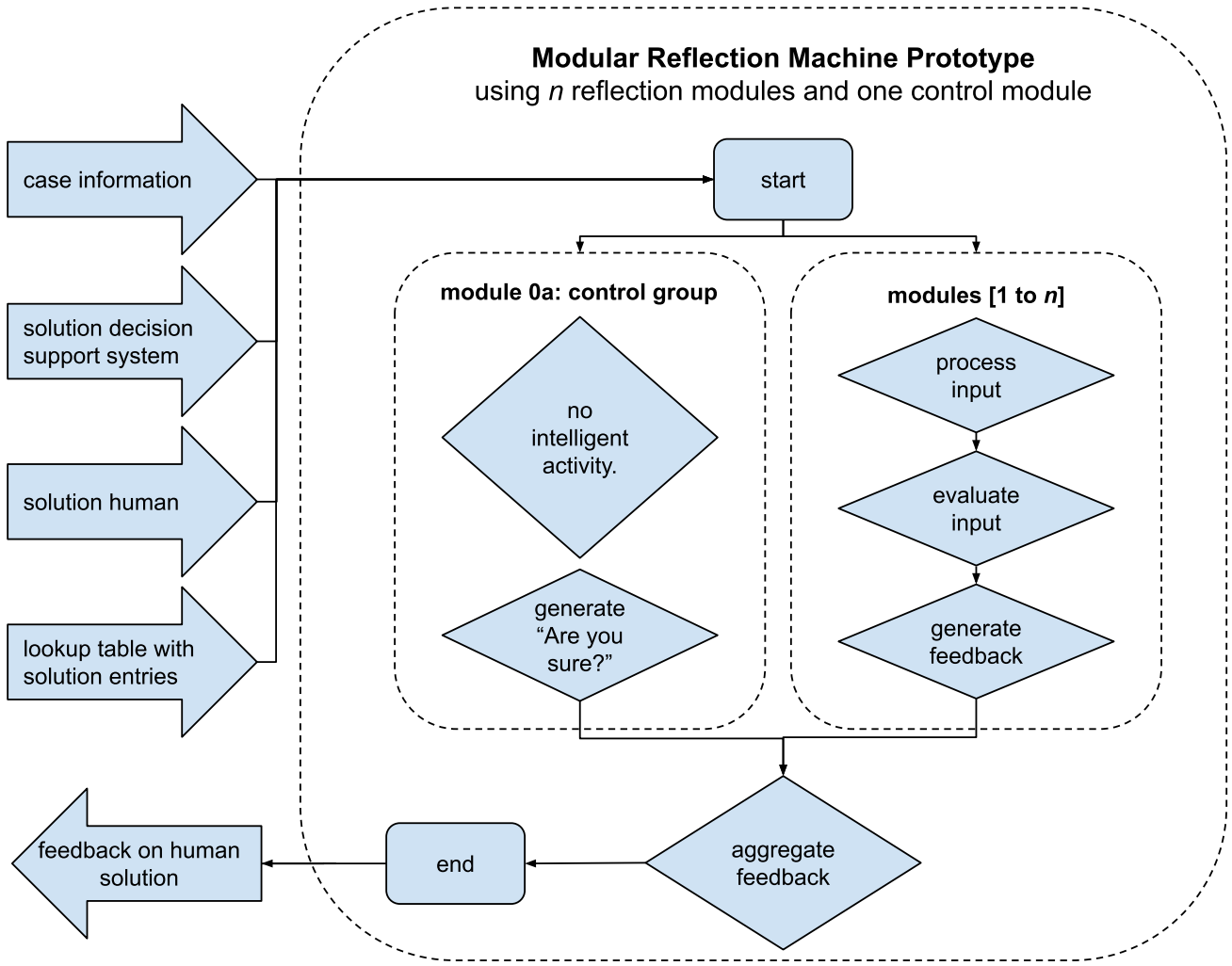

Fig. 1 Specification of a prototype modular Reflection Machine (RM) with module 0a

- An associated feature that helps the aggregator identify similar questions,
- A critiquing weight. The exact meaning of this weight may vary across implementations and problem domains. For this prototype, the weight is a combination of two components. The first is a measure of confidence in the user's solution, where a score of 100% means the module is certain the user's answer is correct, and a score of 0% means the module is certain the user is wrong. The second is a multiplier that is derived from the feature itself. Without this multiplier, all features pertaining to a single solution would hold the same critiquing weight. With the multiplier, features that hold greater significance (i.e., show a greater difference between compared solutions) are more likely to be used for falsification than features of lesser significance.

Finally, the intermediate output of each module is to be aggregated to determine which one question to ask the user. For instance, this can be done by taking the average score of each factor over the different modules. This can be a combination of multiple modules or a single best option as presented by a module individually, depending on what output is most desired / effective. Since this is largely unknown, the weights will be used to determine a single best question to present to the user.

Creating domain knowledge

The domain knowledge available to the prototype was a manually created data set based on the case descriptions, as explained above. The relations between solutions and features were extended by adding data of trusted Dutch self-help websites such as https://richtlijnen.nhg.org/ and https://www.thuisarts.nl/ where necessary. The domain

Table 1 Example of domain knowledge represented in an *n* by *m* matrix

Solution	Feature 1	Feature 2	Feature 3	Feature 4	Feature …	Feature *n*
Answer a	0	1	0	– 1	…	0
Answer b	– 1	– 1	0	0	…	1
…	…	…	…	…	…	…
Answer *m*	0	0	-1	0	…	1

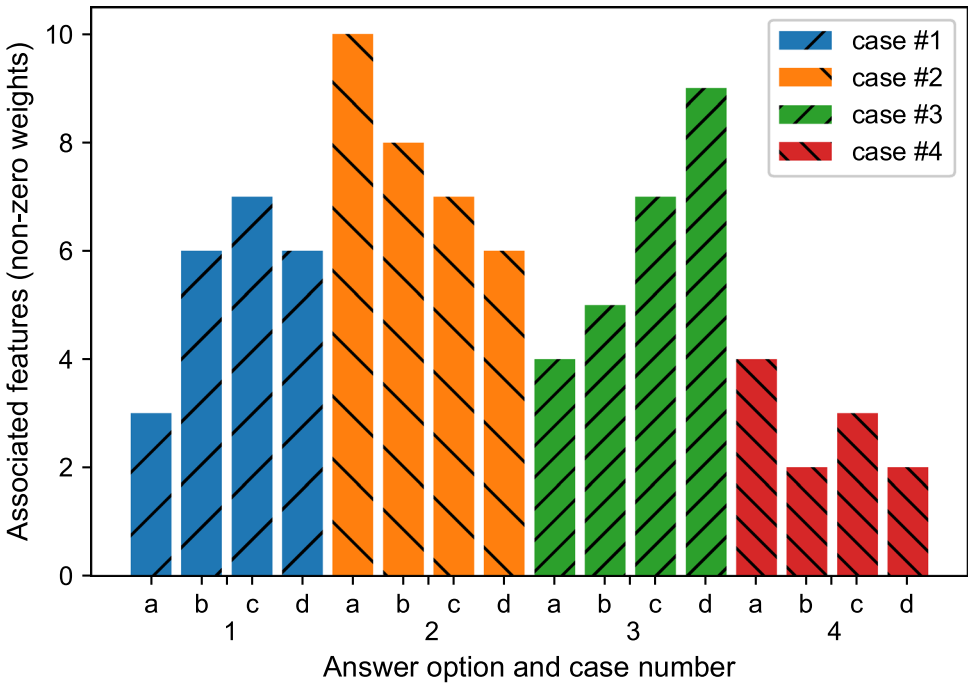

Fig. 2 Distribution of features per case and solution

knowledge can be represented by a simple m by n matrix with columns for each known feature (e.g., symptoms in a medical task) and rows for each known solution. Numerical values record the relation of each feature to each solution. Table 1 shows an example of such a matrix.

Positive numbers represent that a feature is known to be related to a solution, negative numbers show they are known to be unrelated, and zeros denote unknown relationships. The bounds of the weights are arbitrary and will likely not be identical across implementations. They will affect how an RM produces feedback, and the bounds should therefore be consistent across all data sets used with a particular implementation.

The distribution of (either positively or negatively) associated features per case and solution is shown in Fig. 2. This distribution is not perfect but will suffice for the implementation. The DSS suggests, per case, the solution:

(1) D
(2) B
(3) C
(4) D

For this implementation, the values of the matrix were assumed to represent how often a feature *i* is associated with a solution *a*, meaning that the weights of the lookup table can be defined as:

$$weight = \frac{x \leftarrow number\ of\ times\ feature\ i\ was\ associated\ with\ solution\ a}{n \leftarrow number\ of\ recorded\ instances\ of\ solution\ a}$$

Since the actual values of x and n are unknown, the values were simulated to represent the created data set while maintaining the [0, 1] bounds. Each non-zero relation from the data set was assigned randomly from a truncated normal distribution (Burkardt, 2014):

$$\varphi(\mu, \delta, x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{x}}$$

Using an m (mean) of 0.5 and a d (variance) of 0.5, all values taken from the truncated normal distribution will be between our bounds [0, 1] with an overall mean of 0.5.

$$weights \leftarrow \int_{0.0}^{1.0} \varphi(0.5, 0.5, x)$$

This method produces a randomisable set of *simulated* associations between features and solutions, while adhering to the rational *known* relations between features and solutions as shown in Fig. 2.

Determining counter questions

To arrive at a helpful counter question, the RM takes its confidence in the human's decision into account and identifies features that might have been overlooked, misinterpreted, or point towards other solutions. For this, each intelligent module (1, 2 and 3) should compare the user's solution with a part of the RM input. For the correctness module (1), this is the DSS solution. For the consistency module (2) this is the case information. For the differential module (3), this would be a solution generated by the RM based on the case information. Since the purpose of this research was not to generate the best alternate solution, the differential module (3) received the DSS support solution instead[2]. This distribution of inputs means each module receives two groups of features. Either 2 solutions with associated features in the domain knowledge, or a solution and the complete case description which can be matched to the domain knowledge table. For sake of readability, the group of matching features from a case description used by the consistency module (2) will also be referred to as a 'solution'.

To determine the value of a single feature for effective RM feedback, a measure of confidence in the user's solution and a weight to describe the value of a specific feature of the user's solution must be computed. The goal is to find the *factor* that the RM has least confidence in (or alternatively, that best explains why a solution is 'bad'). To obtain a measure of confidence in a solution a, each of its features can be compared to those of a competing solution b. The sum of the absolute difference between each feature results in a total distance between solutions a and b. The total distance divided by the total number of evaluated features $\underline{n}$ gives a degree of dissimilarity between solution a and solution b, and subtracting that from 1 produces a similarity score:

$$distance(a,b) = \sum_{i=0}^{n} abs(a[i] - b[i])$$

$$similarity\,(a,b) = 1 - \frac{distance\,(a,b)}{n}$$

A greater similarity score implies more shared features and can thus be seen as a verdict of confidence in solution a. This method yields a symmetric similarity measure (i.e., the distance from a to b equals the distance from b to a). However, for us it was more interesting to use an asymmetric measure as this allowed for experimentation within modules and different aggregation methods. To obtain this asymmetric similarity score, we only measure the distance for features of which the value for a is known, i.e., not 0:

$$similarity\,(a,b) = 1 - \frac{distance\,(a[where\ a \neq 0], b\,[where\ a \neq 0]}{length\,(a[where\ a \neq 0])}$$

Because the lookup table is bound to [0, 1] and case information mentioning the known absence of a feature is translated to a continuous value between -1 and 0, the maximal distance is $1-(-1)=2$, and the minimal distance is $x-x=0$. For computing the overall confidence, a bound of [0, 2] is not suitable, since the maximal distance between solution a and solution b is then 2 times the number of features, which results in a negative similarity score rather than a minimal score of 0:

$$similarity\,(a,b) = 1 - \frac{2n}{n} = -1$$

Hence, while the distance per feature is bound to [0, 2], it is more practical to bound the total distance to [0, n], which in turn bounds the similarity score to [0, 1]. To achieve this, the distance per feature can be divided by 2 before addition, or the total distance can simply be divided by 2. Finally, by multiplying the similarity value by 100%, a percentile confidence measure is obtained that is both rationally bound to its meaning and useful to determine the value of a module's feedback.

Evaluation

Technical evaluation criteria

As indicated above, a simple way to implement an RM is by having it produce a question like "Are you sure?" or "Did you consider *factor x*?". To meaningfully evaluate the prototype and more effectively differentiate between the performance of each module, 4 additional constraints are detailed below:

(1) The 'better-than-guess' constraint: The confidence scores as produced by the modules should be lower than that of the control modules. If not, there is no rational benefit to presenting the user with the module's feedback rather than that of the control group. Because the machine will be tested on a multiple-choice paradigm with 4 possible solutions, the confidence in an uninformed guess should be 25%. This creates a deci-

[2] In practice, it is assumed that the solution of a good DSS is close to optimal in most cases. Module 3 would be useful in the presumably rare situations where the DSS is suboptimal.

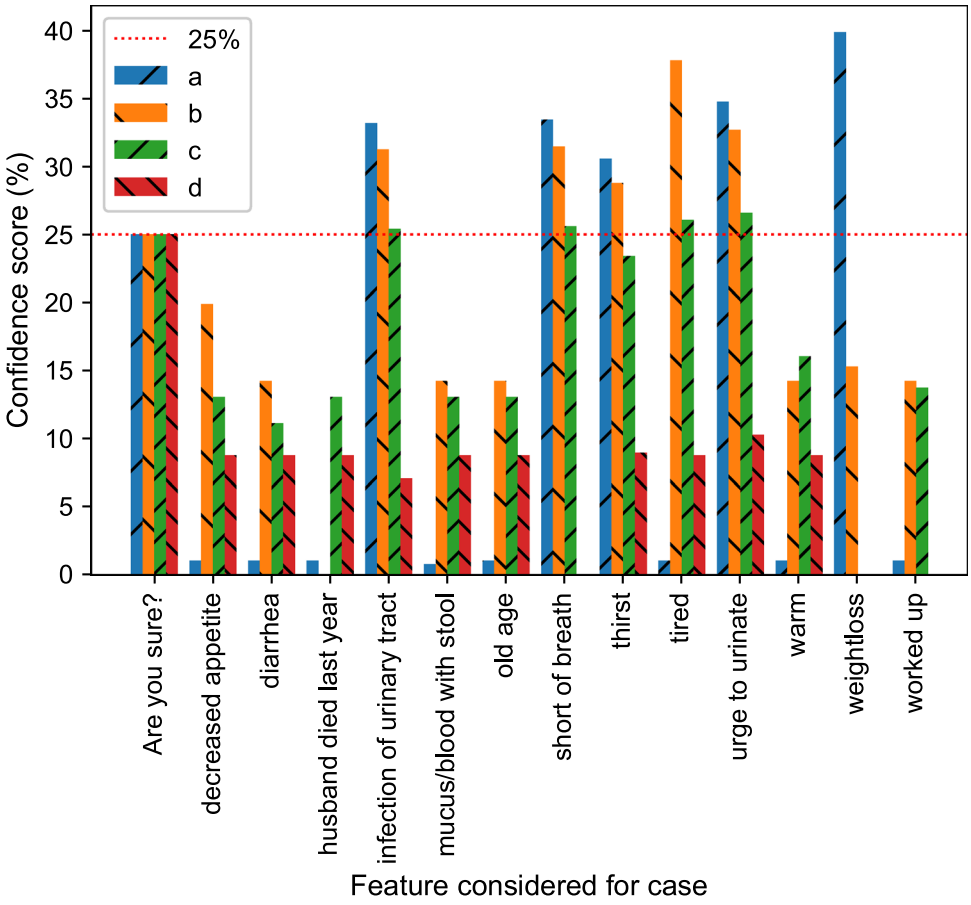

Fig. 3 All features considered as output by the prototype for case 1 with solutions a b c and d

sion threshold of 25% above which the machine will not consider the module's feedback.

(2) The 'weightless' constraint: The confidence scores produced by each module should be rational and meaningful. Theoretically, the scores can be artificially weighed down to pass the 25% threshold. While this may be desirable for some practical scenarios where different modules produce vastly different scores, for this research, each module should aim to pass the threshold without explicit weighting.

(3) The 'varied' constraint: The confidence scores should be sufficiently varied. A homogeneous spread of scores reflects little difference in relevance between the various features. As such, when the produced scores are too homogenous, the RM is not achieving any more intelligent activity than that of the informed control module (0b). Since the module also selects a random but relevant feature, it will ultimately produce feedback that poorly represents the case information or the domain knowledge.

(4) The 'different' constraint: Finally, the feedback that the machine selects should preferably be different depending on the solution proposed by the user.

Notably, the four evaluation criteria above are measures of statistics. They will give a rough estimate of the computational performance of each module based on the limited dataset and limited case studies used. These evaluations do not attempt to evaluate the intelligence or effectiveness of each individual feedback instance produced by the prototype.

Evaluation results

Because the prototype's output is limited to a single question, it is informative to visualise the machine's decision process that determines the output. Figure 3 shows each feature that was considered to be mentioned in the feedback to the user (by one or more of the uninformed control module, correctness module, consistency module and differential module 0a, 1, 2 or 3) for a single multiple-choice decision task. The features here are for case 1[3]. The x-axis shows the feature as it was provided in the domain knowledge, the y axis shows the assigned weight in percent (a confidence score).

[3] For Fig. 3, these features were translated into English. In the domain knowledge the features are described in Dutch.

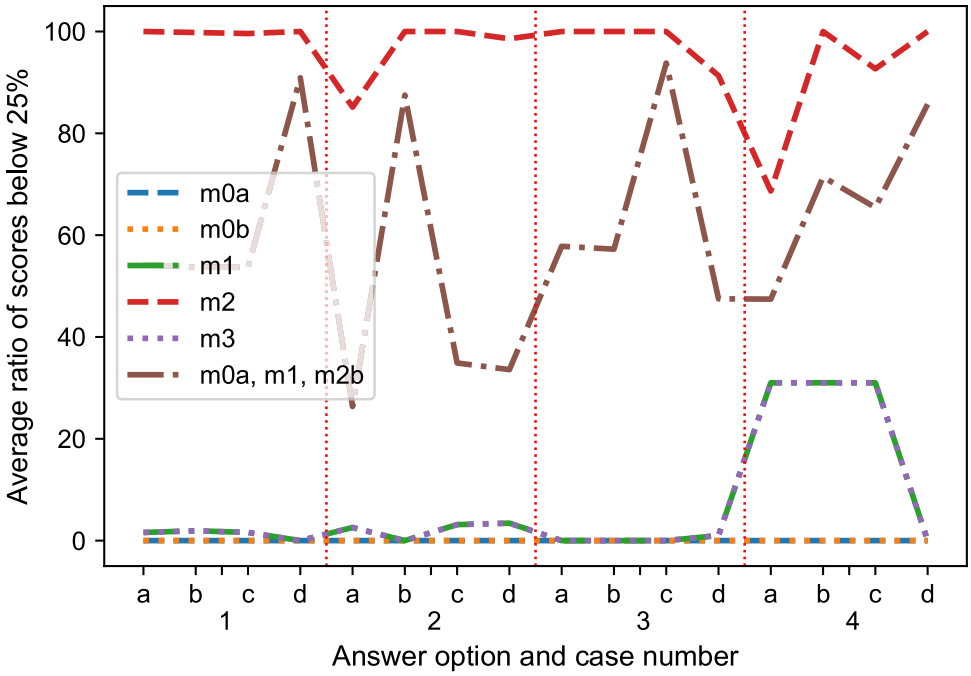

Fig. 4 Ratio of confidence scores below 25%, per module

Different coloured bars show the score of a feature for each of four possible user solutions. The absence of a bar means the feature was not considered as a factor for that particular solution. Features that were not considered for any solution or that had a confidence score of 0% were omitted from the decision process. The decision threshold of 25% is shown as a dotted red line; any score over this line is considered unfavourable compared to the control question 'Are you sure?'.

To evaluate the different criteria described earlier, the produced scores were stored for each case and (combination of) modules. To get a more accurate representation of the behaviour of each module, this process was repeated 50 times. To this end, the weights of the lookup table were randomised multiple times to produce different variations of each case. As such, the graphs below depict the average data for each module and case across all 50 trials.

The 'better-than-guess' constraint

To evaluate the first constraint, the confidence ratio over and under the 25% threshold for the four different tasks and solutions is visualised in Figure 4. Red dotted lines separate the data for each module, where each module was tested on all 4 cases, ordered 1 to 4.

The uninformed and informed control modules (0a and 0b) never score below 25% as they are set to the threshold of exactly 25%. The correctness modules and differential module (1 and 3), which used equivalent implementations, correctly score the same values. Their ratio is not great; only a fifth of their selected features score below the threshold. The consistency module (2), on the other hand, shows consistent low scores, as 9/10th of its scores is under the threshold. The combination of the correctness module and consistency module (1 and 2) logically shows an average of both modules, both producing low and high scores.

The 'weightless' constraint

To evaluate the second constraint and determine the rationality of the produced confidence scores, a similar graph can be looked at for the ratio above and below an arbitrary 50% threshold (see Fig. 5). This informs us of large discrepancies between modules.

The uninformed and informed control modules (0a and 0b) correctly always score under 50%. The consistency module (2) also often scores less than 50%. The correctness and differential modules (1 and 3) show higher overall scores (thus, their ratio below 50% is lower). The combination of correctness and consistency modules (1 and 2) again shows an average of the two, though it is more skewed towards that of the consistency model.

The discrepancy in scores between the consistency module (2) and the correctness and differential modules (1 and 3) can be explained by the number of features they considered (see Fig. 6). Since the correctness and differential modules only compare features between two solutions, the total number of non-zero features is likely lower than that of the consistency model, which compares a solution with all the known features found in the case description.

The case description is built to substantiate features of all four possible user solutions and will therefore always

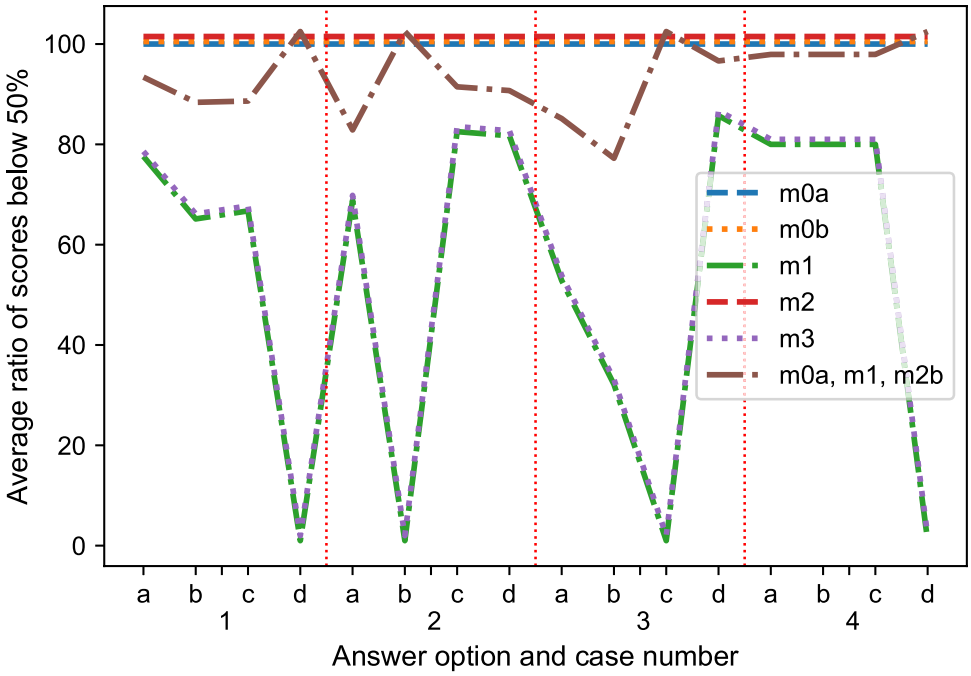

Fig. 5 Average ratio of confidence scores below 50%

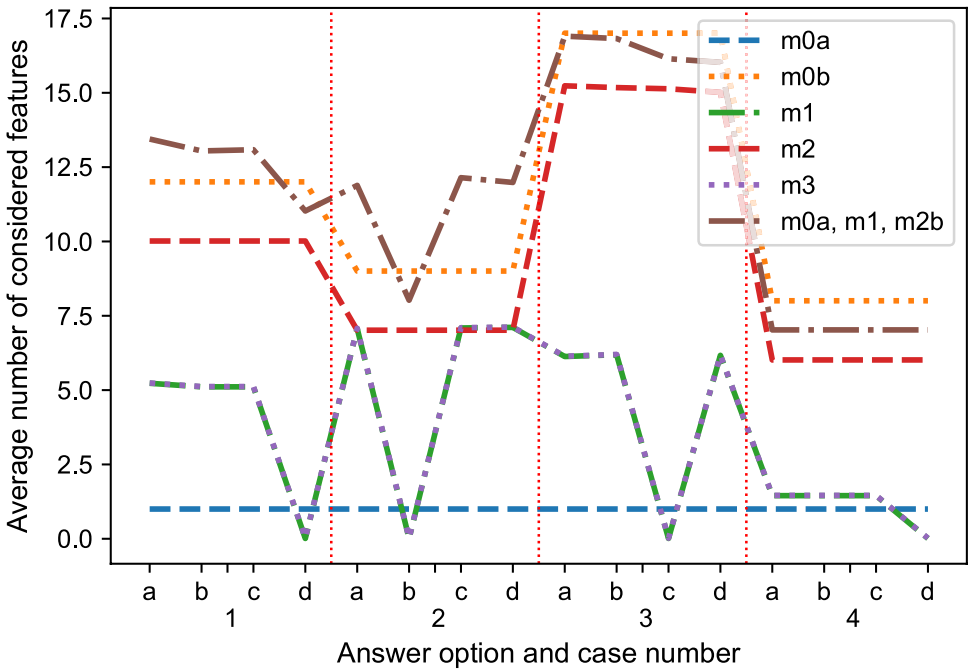

Fig. 6 Average Number of features considered per case

contain features that are irrelevant to or different from the user solution. This leads to overall lower confidence scores. This problem may persist with real-world cases, though weighting of module scores could circumvent this.

The 'varied' constraint

By plotting the standard deviation, we get a rough idea of how the confidence scores of a module are distributed. Figure 7 shows these standard deviations per case and solution, where the red dotted lines again separate modules.

The correctness and differential modules (1 and 3) show greater variation within the confidence scores and also show large variations between the different cases. The consistency module (2) shows less variation, both within solutions and between different cases. The lower variation for the consistency module can partly be explained by the overall lower scores it produces, as discussed earlier. This is backed up

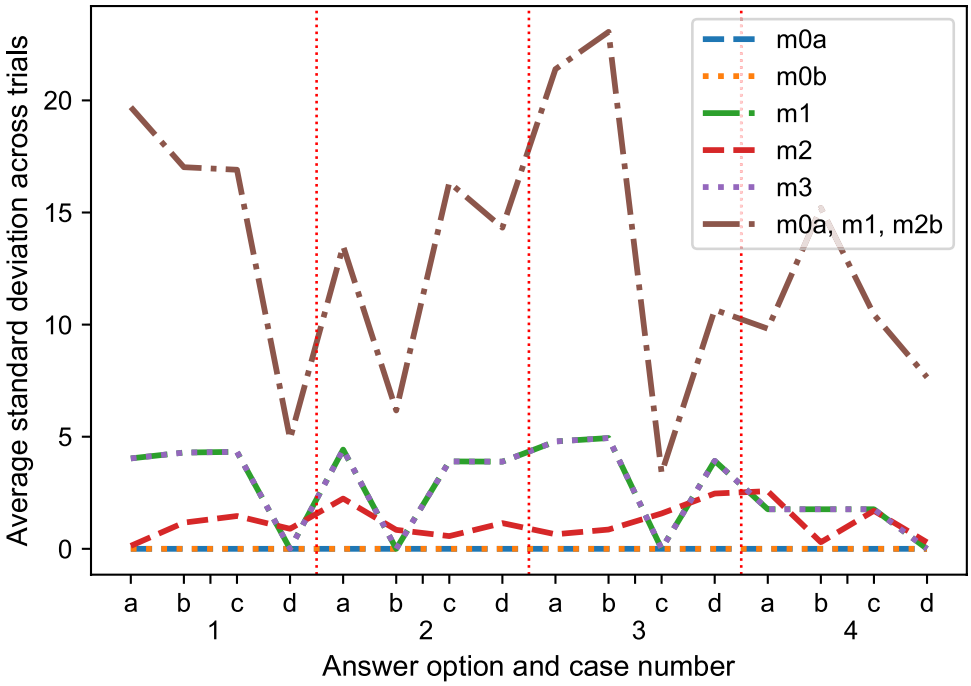

Fig. 7 Average standard deviation of confidence scores

by the combined standard deviations of the correctness and consistency modules (1 and 2), which is much higher than that of either module, indicating they produce vastly different scores.

When looking closely, it appears that there is a clear lower standard deviation when the user submitted solution matches the one selected by the DSS for any trial that included the correctness and differential modules (1 and 3). This is likely because, in the case of a match, these modules do not produce feedback (since the match between the solutions is perfect, there is a distance of 0, thus it creates a confidence of 100%). In the other 3 situations, where the user's solution does not match that of the DSS, variance is much higher. Case 1 and 3 appear to produce higher variance in confidence scores, where cases 1 and 2 appear to produce higher variance.

The 'different' constraint

Ideally, the feedback of the RM depends on the solution proposed by the user. Because the final output of the machine is one question generated using a single feature, and there are 4 solutions the user could enter in this multiple-choice setup, there is a maximal variance of 4 and a minimal variance of 1 (where each answer is the same). This variance is plotted in Fig. 8.

With our implementation, the uninformed control module (0a) always produces the same output, whereas the informed module (0b) could also be implemented to choose a random feature instead, leading to high variance across trials. This would make its output non-deterministic, which makes it hard to compare it to the other modules, which are deterministic.

The correctness and differential modules (1 and 3) produce more varied output, varying between 2 and 4 different messages.

The consistency module (2) shows more promising variation in answers, though it also does not achieve output that is perfectly unique to the user's solution except for case 4. By combining the uninformed control module, the correctness module and the consistency module (0a, 1 and 2), variance equal to that of correctness and differential modules (1 and 3) is achieved.

To sum up, the performance of the modules varies. The RM aims to make the user rethink, or reflect on, their decision to avoid them leaving the decision entirely up to the DSS. The proposed implementation of the correctness module (1) can only act when the decision of the user is already different than that of the DSS. The consistency module (2) does not suffer from this problem, as its proposed implementation can detect problems with the GP's decision even if that decision is the most optimal or obvious solution. The differential module (3) is conceptually a solution to the shortcoming of the correctness module, as in case of a match it can still use an alternate solution for its question. This will be especially useful in the rare cases where the DSS solution is suboptimal. Consider the example of the child with abdominal pain where the GP agrees with the DSS: while the correctness module (1) would not produce questions, the other modules could compensate,

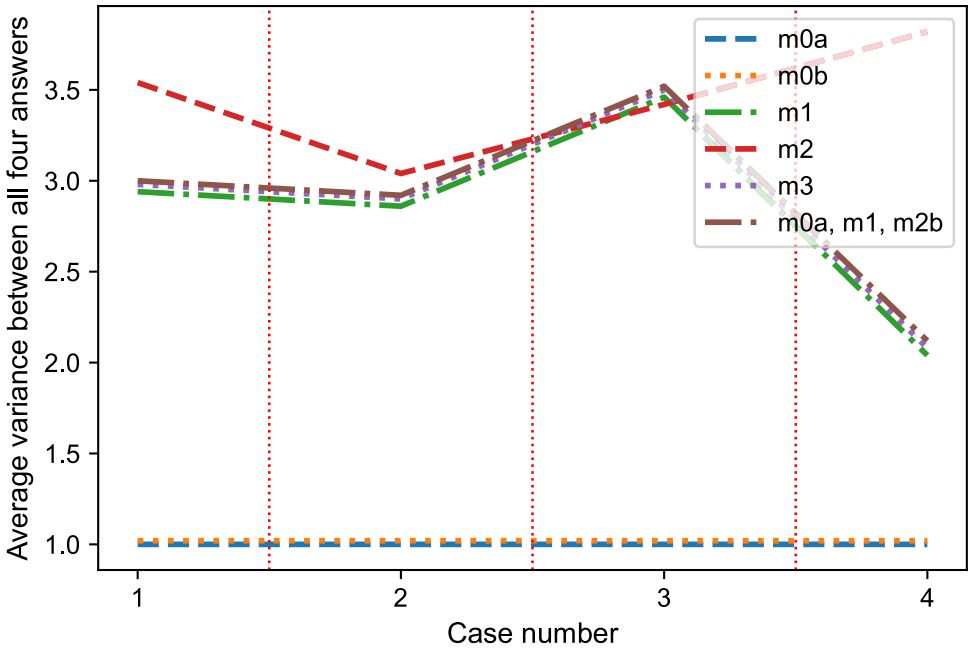

Fig. 8 Average variance of unique outputs by the modules

e.g. the differential module (3) may ask about the presence of bruises, and the consistency module (2) may note that the child does not have a fever.

As such, we suggest that the different modules, or methods of producing counter questions, may work best when combined. This is also backed up by the higher variance in suggested feedback and the higher variance in the produced confidence when multiple modules are used in tandem. While the effectiveness and relevance of the produced feedback are uncertain, the prototype's results indicate that it is easy to produce these questions with minimal data.

Discussion and conclusion

This article has proposed the concept of RMs to increase meaningful human control over DSS. We have presented and evaluated a prototype to demonstrate the technical feasibility of RMs. The prototype shows promise as a way to implement a counter-thinking machine. It turns out that based on the modular approach, relevant questions can be determined and ranked by confidence in the human expert's solution based on minimal data that is generally readily available. The modular approach allows different forms of critics to compete and cooperate to achieve useful variance in the machine's output.

A primary goal of an RM is to reduce passive decision-making when using a DSS. In this sense, the RM is not concerned with the quality of the resulting decision but with the quality of the decision-making process. Nevertheless, we should not forget that the accuracy of the resulting decisions will be an important criterion for the real-world adoption of RMs. A key question is thus how the accuracy of decisions changes with the introduction of an RM. The final outcomes of a DSS-supported decision-making process can be affected by the RM in a variety of ways: For instance, based on the RM feedback, the user might change an incorrect decision into a correct decision - or change an incorrect decision into a correct decision. Likewise, the RM might also cause the user to change their decision towards the output of the DSS (which is expected to be correct most of the time) or motivate them to deviate from the DDS output. In the example about abdominal pain that we described, a RM question like 'how does the child experience school?' might cause the GP to find out that the child is suffering from stress at school. This could help them to avoid a kind of 'tunnel vision' following the DSS's conclusion that the child has the flu. In our vision of RMs, the RM feedback increases the human's involvement in the decision process without *systematically* steering them away from or towards the DSS output. Ideally, the presented RM question and the following deliberation process reassure users that they made the right decision when this is the case, and points them towards a more likely solution when they have overlooked or misinterpreted information. For example, the machine may ask whether the child has had fever in the past days, and if the parents confirm so, the GP can conclude that their instinct, and the suggestion of the DSS, were correct. But if the parents note the child has not suffered from fever recently, the GP can consider alternative diagnoses.

While RM questions can steer the user towards or away from the output of the DSS, whether or not, or to what degree, this happens depends on the actual RM

implementation. In our prototype, we present questions after *each* decision, thus prompting additional reflection independently of whether the user has accepted the DSS decision or not and independently of how confident the RM is in the user's decision. At the same time, our RM aims to produce *helpful* questions that point out factors that, e.g., might have been overlooked. The resulting questions thus might nudge users towards statistically more likely answers/DSS answers. In this way, the RM involvement could increase both the quality of the decision-making process and the quality of the decisions. While we have presented one specific prototype in this paper, RMs can take many different forms, and their effectiveness likely depends on design choices. We thus suggest further exploring the design space of RMs. Furthermore, the system presented in this article could be scaled up to solve more open-ended problems.

The RM could be a computationally efficient way to stimulate self-reflection, critical thinking and increase meaningful human control over the rapidly developing DSSs in various domains, thereby increasing the possibility that humans genuinely remain 'on' the loop. However, the effectiveness of the RM and its questions is yet to be determined, as is the willingness of human experts to collaborate with such a machine, especially in real-world applications and settings. Therefore, future work should empirically evaluate the concept. As part of this, it would be interesting to measure the user's sense of agency (Lauermann & Karabenick, 2013) and reliance on external systems, such as the DSS and RM. This could be done with statements such as "I made my diagnosis completely independently", "The diagnosis was a conscious decision", "The diagnosis support system has influenced my answer", "The question from the RM has influenced my answer", "I am responsible for what my actions might bring about" (based on Tapal et al., 2017).[4]

When evaluating the real-world value of RMs in the future, it is crucial also to consider other essential criteria for the decision-making process, such as the quality of the decisions and the efficiency of the process. Unfortunately, tensions between values such as human control, efficiency, accuracy, and responsibility might emerge, and trade-offs might have to be made. Because of these value tensions, we suggest pursuing a value-sensitive design approach (Friedman et al., 2013) to researching decision making with DSSs and RMs.

Finally, RMs are not the only approach to the underlying problem of fostering meaningful human control over DSS. For example, changes to the design of DSSs themselves might also improve human involvement. Likewise, other complementary approaches, such as introducing design friction in the user interface (Cox et al., 2016) to foster more reflective behaviour, can be explored to ensure meaningful human control over DSSs. In this paper, we have introduced and explained the basic elements and functions of an RM, but, like all machines, RMs will need to show their utility in practice.

[4] A pilot following this approach with 20 participants has already been conducted. Unsurprisingly, given the small number of participants, no clear trends could be discerned regarding users' sense of agency.

Author contributions Modelling & Implementation RM & text: NAJC. Case construction & pilot study RM & text: RJMvE. Feedback & text: HKS. Idea & background RM & text: WFGH

Funding The authors did not receive support from any organization for the submitted work.

Data availability The datasets generated during and/or analysed during the study are available in the anonymized Git repository: https://anonymous.4open.science/r/Reflection-Machine-Prototype-21B3/README.md.

Code availability The code produced during the study is available in the anonymized Git repository: https://anonymous.4open.science/r/Reflection-Machine-Prototype-21B3/README.md.

Declarations

Conflict of interest The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

Budgell, B. (2008). Commentary guidelines to the writing of case studies. *The Journal of the Canadian Chiropractic Association, 52*(4), 199–204.

Burkardt, J. (2014). *The Truncated Normal Distribution* (pp. 1–35). Department of Scientific Computing Florida State University.

Chen, Y., Argentinis, E., & Weber, G. (2016). IBM Watson: How cognitive computing can be applied to big data challenges in life sciences research. *Clinical Therapeutics, 38*(4), 688–701. https://doi.org/10.1016/j.clinthera.2015.12.001

Cox, A. L., Gould, S. J., Cecchinato, M. E., Iacovides, I., & Renfree, I. (2016). Design frictions for mindful interactions: The case for microboundaries. In *Proceedings of the 2016 CHI conference extended abstracts on human factors in computing systems* (pp. 1389–1397).

Dechesne, F., Dignum, V., Zardiashvili, L., & Bieger, J. (2019). *AI and ethics at the police: Towards responsible use of artificial intelligence in the Dutch Police*. Leiden University.

de Santoni, F., & Mecacci, G. (2021). Four responsibility gaps with artificial intelligence: Why they matter and how to address them. *Philosophy and Technology*. https://doi.org/10.1007/s13347-021-00450-x

de Santoni, F., & van den Hoven, J. (2018). Meaningful human control over autonomous systems: A philosophical account. *Frontiers Robotics AI, 5*(FEB), 1–14. https://doi.org/10.3389/frobt.2018.00015

Eysenck, M. W., & Keane, M. T. (2002). *Attention and performance limitations. Foundations of cognitive psychology: Core readings*. MIT Press.

Friedman, B., Kahn, P. H., Borning, A., & Huldtgren, A. (2013). Value sensitive design and information systems. In K. E. Himma & H. T. Tavani (Eds.), *The handbook of information and computer ethics* (pp. 55–95). Hoboken, NJ: Wiley.

Grissinger, M. (2019). Understanding human over-reliance on technology. *Pharmacy and Therapeutics, 44*(6), 320–321.

Kieffer, S. (2017). Representative design in user experience evaluations. *AIS Transactions on Human-Computer Interaction, 9*(2), 149–172.

Kruse, C. S., & Ehrbar, N. (2020). Effects of computerized decision support systems on practitioner performance and patient outcomes: Systematic review. *JMIR Medical Informatics, 8*(8), 1223–1238. https://doi.org/10.2196/17283

Lauermann, F., & Karabenick, S. A. (2013). The meaning and measure of teachers' sense of responsibility for educational outcomes. *Teaching and Teacher Education, 30*, 13–26. https://doi.org/10.1016/j.tate.2012.10.001

Liu, H. Y. (2018). The power structure of artificial intelligence. *Law, Innovation and Technology, 10*(2), 197–229. https://doi.org/10.1080/17579961.2018.1527480

Matthias, A. (2004). The responsibility gap: Ascribing responsibility for the actions of learning automata. *Ethics and Information Technology, 6*(3), 175–183. https://doi.org/10.1007/s10676-004-3422-1

Mecacci, G., & de Santoni, F. (2020). Meaningful human control as reason-responsiveness: The case of dual-mode vehicles. *Ethics and Information Technology, 22*(2), 103–115. https://doi.org/10.1007/s10676-019-09519-w

Merritt, S. M., Ako-Brew, A., Bryant, W. J., Staley, A., McKenna, M., Leone, A., & Shirase, L. (2019). Automation-induced complacency potential: Development and validation of a newscale. *Frontiers in Psychology, 10*(FEB), 1–13. https://doi.org/10.3389/fpsyg.2019.00225

Phillips-Wren, G., Mora, M., Forgionne, G. A., & Gupta, J. N. D. (2009). An integrative evaluation framework for intelligent decision support systems. *European Journal of Operational Research, 195*(3), 642–652. https://doi.org/10.1016/j.ejor.2007.11.001

Robbins, J. E. (1998). Design Critiquing Systems. Technical Report UCI-98-41.

Roig, A. (2017). Safeguards for the right not to be subject to a decision based solely on automated processing (Article 22 GDPR). *European Journal of Law and Technology, 8*(3), 1–17.

Tapal, A., Oren, E., Dar, R., & Eitam, B. (2017). The sense of agency scale: A measure of consciously perceived control over one's mind, body, and the immediate environment. *Frontiers in Psychology, 8*, 1552. https://doi.org/10.3389/fpsyg.2017.01552

van der Stigchel, B., van den Bosch, K., van Diggelen, J., & Haselager, W. (submitted) Intelligent decision support in medical triage: are people robust to biased advice? *Journal of Public Health*.

van der Stigchel, B. (2021). *Resilience towards bias in artificial agents: Human-Agent team performing a triage task*. MSc Thesis, dpt. of Artificial Intelligence, Radboud University, Nijmegen, The Netherlands.

van Eerdt, R. (2021). *Falsification machines in medical decision making*. Radboud University.

Wachter, S., Mittelstadt, B., & Floridi, L. (2017). Why a right to explanation of automated decision-making does not exist in the General Data Protection Regulation. *SSRN*. https://doi.org/10.1609/aimag.v38i3.2741