Improving Learning Environments: Avoiding Stupidity Perspective

Radek Pelánek and Tomáš Effenberger

***Abstract*—Research in learning technologies is often focused on optimizing some aspects of human learning. However, the usefulness of practical learning environments is heavily influenced by their weakest aspects, and, unfortunately, there are many things that can go wrong in the learning process. We argue that in many circumstances, it is more useful to focus on avoiding stupidity rather than seeking optimality. To make this perspective specific and actionable, we propose a definition of stupidity, a taxonomy of undesirable behaviors of learning environments, and an overview of data-driven techniques for finding defects. The provided overview is directly applicable in the development of learning environments and also provides inspiration for novel research directions and novel applications of existing techniques.**

***Index Terms*—Computer-aided instruction, student modeling, defects, data-driven techniques.**

I. Introduction

Today, artificial intelligence techniques are widely used to support human learning. The focus, particularly of research papers, is on optimizing the learning experience. This is, however, very hard. Learning is a complex process, and it is difficult to formulate clear, universal "best practices" in education. At the same time, there are many things that can go wrong, for example, lack of motivation, lack of trust, or missing prior knowledge. For any teaching agent, human or artificial, it is nontrivial to avoid all potential pitfalls. Thus, we believe that in the development of learning environments, it may be more useful to focus primarily on avoiding stupidity rather than achieving intelligence.

Our aim is to make the avoiding stupidity perspective actionable. To do so, we consider the inverted formulation of the aims of a learning environment: What makes a poor learning experience? How to thwart student learning? These questions lead us to consider possible ways how learning environments could fail. Once we carefully map potential defects and undesirable behaviors of learning environments, we can use automated techniques to detect them. We are specifically interested in hidden defects that manifest themselves only in specific parts of the system or only for some subset of students. Such defects cannot be reliably detected by simple testing of the system. However, learning environments collect extensive data, and it is possible to use data-driven techniques to uncover these defects. The main focus of this work is thus on the iterative improvement rather than the initial design of a learning environment.

We focus on detecting behaviors that are clearly wrong or missing behaviors that are clearly missed opportunities. Examples of clearly wrong behaviors are accepting an incorrect answer with praising feedback, assigning items that are too difficult for a student, or showing a feedback message that is not appropriate for the target audience of the learning environment. An example of missed opportunity is a situation where some content is inadvertently inactive or hard to find, and thus a simple change of navigation may have a nontrivial positive impact. Another example is a missing feedback message for a very common error, particularly when other, less common errors have feedback messages.

In an idealized setting, a learning environment that is presented to students does not contain any defects. We would like to avoid problems by good, research-based design and to find all remaining problems by testing before shipping the product to users. However, when we take into account real-world constraints, such an approach is not necessarily optimal. The development of an error-free system and content is very expensive. With a limited budget, we face the choice between developing a very well-designed and well-tested environment with limited functionality and content and developing an environment with rich functionality and wide content, which, however, may contain some defects. This is, of course, not a binary choice but a choice along a continuous spectrum. The appropriate choice depends on the setting. For example, a different choice is suitable for a project for teaching mathematics in English (which can potentially have a very large audience and project budget) and a learning environment for a specialized topic in a language with a small number of native speakers (which necessarily has a small budget). In many settings, it is more rational to aim at an environment with wider coverage and to systematically employ methods for the detection of defects.

The main aim of this work is to highlight the perspective of avoiding stupidity in the context of the development of learning environments and to make this perspective actionable. We focus on learning environments that provide students with practice opportunities and collect data on student performance (which can be used for automated detection of defects). We clearly distinguish among defects, undesirable behaviors of learning environments, and their negative effects on students. We discuss the main types of these negative effects and propose a taxonomy of undesirable system behaviors. We focus on hidden defects and provide an overview of data-driven techniques for finding them, together with specific ex-

Manuscript received June 21, 2021; revised December 6, 2021; accepted TBD. Date of publication TBD; date of current version December 6, 2021. *(Corresponding author: Radek Pelánek.)*

The authors are with the Faculty of informatics, Masaryk University, 602 00 Brno, Czech Republic (e-mail: xpelanek@fi.muni.cz, tomas.effenberger@mail.muni.cz).

amples. The presented taxonomy and overview of techniques can serve as a useful tool in the practical development of learning environments. For researchers, it provides impulses for the development of new techniques for the detection of defects and inspiration for the reinterpretation of some existing techniques. For example, student modeling techniques are typically motivated by the need to provide adaptive instruction; our discussion shows that they may also be useful for defect detection.

II. Our Approach to Avoiding Stupidity

We start by providing an overall overview of the approach that we propose. We define what we exactly mean by the term *stupidity* and discuss the distinction between stupidity as a system action and its effects on students. We also describe methods that we used to reach specific lists of defects and their negative effects, which are then described in the following sections.

A. Defining Stupidity

In order to make the avoiding stupidity perspective actionable, we need to clarify what we are actually trying to avoid. We define *stupidity* as a behavior of a learning environment that is clearly wrong or missing behavior that is clearly a missed opportunity. *Behavior* is any system action that is visible from the student's perspective; *missing behavior* is an unperformed action. *Wrong* means "negatively impacting long-term learning." This negative impact can be caused in different ways, for example, by undermining motivation, trust, or the effectiveness of learning. A *missed opportunity* is an action that would significantly improve long-term learning at a low cost.

A key part of the definition is *clearly*. Long term impact of actions in a learning environment is often hard to establish. We are interested in behaviors that are *clearly wrong* (or clearly a missed opportunity) once detected; that is, once the specific aspect of system behavior is highlighted, most human experts would agree that it is problematic. Additionally, it should be quite clear what type of action to take to correct it. That does not mean that stupidities are trivial mistakes or that people who caused them are stupid. The *clear* aspect of stupidities often involves the use of hindsight bias. For example, once experts are shown difficulty statistics for items, they can clearly see problems that they would miss without these statistics [1].

The term stupidity, even when technically defined as above, may be perceived pejoratively. In the following, we thus use the term *undesirable behavior* as a synonym for stupidity in the above-given sense.

B. Defects, Undesirable Behaviors, and Their Effects

In the discussion of deficiencies of learning environments, it is useful to clearly distinguish system actions and student reactions (effects of the action). It is easy to conflate them, especially with ambiguous terms like "demotivation," which can be used to describe both a system action (e.g., assigning a too difficult task) and the student reaction (e.g., frustration and decreased self-confidence).

The undesirable behaviors, which we aim to describe and detect, are performed by the system. These behaviors are wrong insofar as they have negative effects on the students—these negative effects are the reasons why certain actions are wrong. In most cases, we cannot use these negative effects to detect defects directly. Most of them are complicated psychological constructs. They can manifest themself only after a long time, often as a cumulative reaction to multiple actions with nontrivial interaction with the student's particular situation. For example, presenting repetitive items may contribute to disinterest, but if the student is just doing assigned homework in the learning environment, the disinterest may only manifest in the student's decisions outside the environment.

To clearly distinguish between causes and effects, it is useful to consider terminology used in software engineering for describing software anomalies. The IEEE standard [2] distinguishes defects, failures, and problems. A *defect* is a specific deficiency of a product (e.g., a mistake in the code). A defect can cause a *failure*, which is an event when a product does not perform required actions. A failure typically leads to a *problem*, which is a negative situation experienced by people using the product. We utilize this approach, using the term *undesirable behavior* instead of failure[1]. Also, instead of the generic term problem, we use the more specific phrase *negative effect on students*.

Fig. 1 shows a diagram that depicts the causal relations between defects, undesirable behaviors of learning environments, and negative effects on students. Defects are specific problems in the realization of the learning environment (e.g., texts of tasks, parameter settings, implementations of algorithms). These defects manifest themselves as observable undesirable behaviors of the system, having a direct impact on students' affect and cognition processes. These direct impacts are typically unobservable; cumulatively, they lead to potentially long-term changes in student state (demotivation, incompetence). In the end, they result in observable behavior of students: nonuse, misuse, or poor performance.

The figure is, of course, simplified. We do not claim that it is a complete depiction of all negative consequences and their relations. The main aim of the figure is to clarify the main notions and to provide guidance in our search for defects that can be corrected.

In the following sections, we discuss the elements in this figure in more detail. Section III provides a discussion of the negative effects on students and their relations. Section IV provides a detailed taxonomy of undesirable behaviors, that is, the specific system actions that cause these negative effects.

C. Related Work

Our approach in this work is unique in the attempt to explicitly define the notion of stupidity. The overall perspective, however, is closely connected to several other threads in the current research. One class of related research is the

[1]As opposed to many other software systems, learning environments typically do not have a clear specification since learning is hard to formally specify. Without clear specification, it is too strong to use the term "failure."

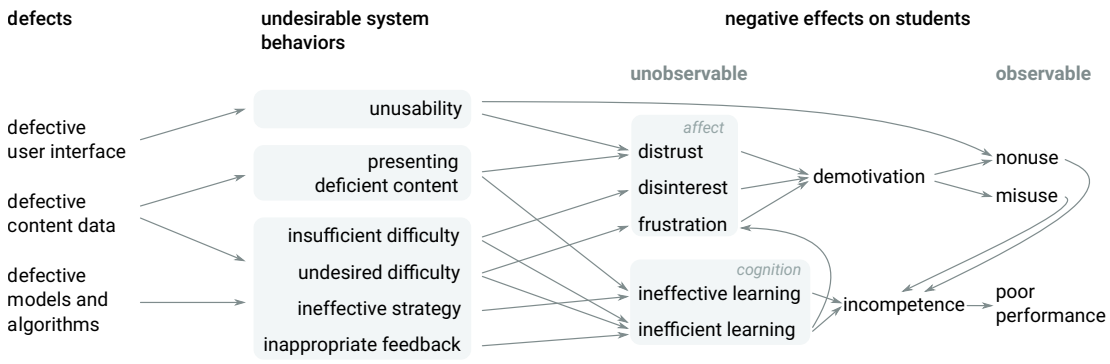

Fig. 1. Overview of the avoiding stupidity perspective to improving learning environments.

design loop adaptivity [3] and *closing the loop* studies [4]. These works are typically formulated in terms of "improving" learning environments rather than finding stupidities but share the aspect of identifying a weak spot of a system and changing it. The data-driven detection of defects can be seen as a special case of *human-in-the-loop* and *intelligence augmentation* [5]. A specific proposal for the use of intelligence amplification in the development of learning environments is Baker's proposal for "Stupid tutoring system, intelligent humans" [6]. Baker uses "stupid" mostly as a synonym for "simple," whereas we use "stupidity" to denote mistakes, so the specific focus of our work differs. However, there is high-level agreement on intelligence amplification and the use of many simple techniques (rather than one complex one). In the closely related field of recommender systems, McNee *et al.* [7] formulated the "don't look stupid" principle—the importance of avoiding poor recommendations that can undermine user trust in the recommender system.

From an engineering perspective, the focus on stupidities is related to the focus on the *weakest link*. A specific example in education is the "What's most broken?" study [8]. In software engineering, there is traditionally a strong focus on debugging and testing—accepting the reality that people make mistakes and developing systematic techniques for finding these mistakes [9].

D. Method

In the following sections, we provide overviews of specific negative effects, undesirable behaviors, defects, and methods for detecting them. These issues are covered in the literature under many names, often just as a sidenote in research papers that primarily focus on different topics. It is thus not feasible to perform a systematic review based on keyword search. Instead, we used an iterative process to construct the presented overviews. This process involved several interconnected steps:

- We started with our own extensive experience with the development of learning environments, cataloging specific problems that we have encountered and techniques that we have used for their detection.
- We identified classic pedagogical works relevant to the development of learning environments and used the inverted approach, that is, formulation of violations of established pedagogical principles.
- We performed manual clustering of similar undesirable behaviors. For identified clusters, we tried to formulate additional similar undesirable behaviors.
- For drafted formulations of undesirable behaviors, we searched for research works that address the specific issues.
- We considered techniques widely used in educational data mining and artificial intelligence in education (e.g., student modeling) and tried to find their uses for the detection of defects and undesirable behaviors.

The core focus of the process was the taxonomy of undesirable behaviors, which is presented in Table I. We performed several iterations of the above-given steps during which we refined the taxonomy.

III. Negative Effects on Students

In this section, we describe the negative effects and relationships among them. Having a good mental model of the negative effects expands our notion of "clearly wrong," helping us more accurately judge which system actions are likely to cause which negative effects, including the indirect and unobservable ones. As this is a vast and complex topic, we necessarily need to greatly simplify the matter. We provide pointers to the relevant literature that offers a deeper coverage.

The literature is also helpful in identifying possible problems—actions that were previously demonstrated to cause these negative effects. Frequently, it is useful to take an inverted formulation of previous research results, that is, to think about how we can break the recommendations or which actions could cause a given negative effect.

We gradually build a terminology to label various effects (and groups of effects), attempting to clearly distinguish between actions and effects. This separation facilitates clearer thinking about undesirable behaviors of systems and students and guides the creation of the taxonomy by grouping undesirable behaviors with similar effects. A shared concise vocabulary also streamlines the discussion in the next chapter.

A. Distrust

Distrust refers to not believing that the system is a good tutor. It is a negative affect towards the learning environment and demotivates using it.

1) Causes: Wang [10] suggests several behaviors that could undermine trust in learning environments: privacy and security violations, unreliable access, unusability of user interface, and poor quality of learning materials. Research from real classrooms gives further inspiration for other negative behaviors. The results differ widely depending on the context, but there are a few frequently reported teachers' misbehaviors: confusing or boring lectures, unfair assessment, poorly organized materials, and teacher's condescending or indifferent attitude toward students, for example, insulting them, treating them like children, and not answering their questions [11], [12].

2) Consequences: In real classrooms, distrust in the teacher's competence has been repeatedly identified as one of the main sources of demotivation [11], [12]. Similar consequences are likely in learning environments as well, most critically for students with disabilities [10]. Baker *et al.* [13] demonstrated that distrust can lead to gaming the system, a form of misuse. In the related field of recommender systems, trust is considered a critical property, especially for the retention of new users [14], [15]. Indeed, showing just one nonsensical recommendation can lead to distrust in the system even if the other recommendations are perfect [7].

B. Disinterest

Disinterest refers to not believing that engaging would be enjoyable. It is a negative affect towards a certain activity or topic, and it reduces intrinsic motivation to engage in the activity or learn the topic [16]. From the stupidity perspective, it is useful to distinguish between personal and situational interest [17]. Personal interest is much more stable, takes a lot of time to develop [18], and is unlikely to be undermined by a single undesirable behavior. Situational interest, in contrast, is fleeting and highly impacted by the features of the specific activity.

1) Causes: Disinterest is caused by a lack of challenge, novelty, or choice [16]. Several studies looked specifically at features of educational texts [19], [20]. An anti-problem view of their findings suggests that situational disinterest can be promoted by poorly organized, incoherent, vague, dull texts, which are irrelevant or insufficient for the task at hand. These findings are supported by studies in real classrooms; students often report uninteresting materials and monotonous activities as demotivators [21]–[23].

Similar to the situational interest is the concept of flow [24]. These two constructs highly correlate [25], and flow can be viewed as a more extreme version of the interest [26]. The flow theory suggests three ways to reduce interest: unclear goal, inappropriate difficulty, and insufficient feedback.

Malone created a taxonomy of features that promote intrinsic motivation in learning environments [27]. In addition to the explicit goal, appropriate challenge, and frequent performance feedback, the taxonomy also lists the appropriate level of control, support of sensory and cognitive curiosity, endogenous story related to the practiced topic, social recognition, and an option to compete or cooperate. Similar taxonomies of motivating features can be found in gamification literature [28]. As noted by Malone, it is not necessary—and possibly even not desirable—to incorporate all motivating features into a single activity. One missing motivating feature is not an issue; their complete omission is undesirable.

2) Consequences: Situational interest supports attention, persistence, and motivation [16] and is even related to the quality of learning and deep understanding [17], [18]. Omission of features that elicit situational interest is especially problematic for novices and low-performing students [18], [26].

C. Frustration

Frustration refers to not believing that mastery of the topic is achievable. It is a negative affect targeted mostly towards the students themselves.

1) Causes: According to flow theory [24], a general cause of frustration is an excessive difficulty, which results in an unproductive effort, failure, and feedback about one's incompetence. Not all failures are frustrating, and some are useful for long-term learning [29]. But some failures are unproductive and frustrating; these may be caused by unclear instructions [11], [30] or insufficient guidance [31]. Other failures are productive but unnecessarily frustrating; these might be caused by comparison to better students or inappropriately harsh evaluation criteria (e.g., when a student does not receive any credit for a multi-step item after a single mistake).

2) Consequences: Frustrating experiences are accompanied by negative emotions and lower self-confidence, especially for beginners [21]. In extreme cases, they can even lead to learned helplessness. The behavioral consequence of frustration may be nonuse since students avoid potential failures to protect their self-worth by not even trying [32]. If the students have to use the system, frustration can lead to misuse, such as gaming the system [13].

D. Ineffective Learning

Learning is ineffective when the learning goal is inappropriate. In contrast to the distrust, disinterest, and frustration, the problem here is not in missing motivation but in the misdirected effort. Students might even experience positive emotions but at the cost of actual learning.

1) Causes: An extreme case of ineffective learning is delivering false information (e.g., factual errors in learning materials or wrong "correct answers"). More subtle is missing feedback, which can also contribute to misconceptions and overconfidence [33]. A less severe case is not learning something important about the topic, either because it is missing in the learning materials or possibly because mastery was declared too early. Or the coverage is sufficient, but the depth is not; for example, only simple recognition is practiced, not any higher-level skills. Ineffective learning also includes spending time on a topic that the student already knows well. This can happen due to misdirected gamification. For example, when the learning environment rewards students for solving tasks even when they are trivial for them.

2) Consequences: Some forms of ineffective learning promote misconceptions. However, ineffective learning does not always lead to outright false knowledge; knowledge that is incomplete, shallow, or fragile is often the case. Ineffective

learning can also contribute to the illusion of knowledge, that is, students thinking they are competent while they are not [34].

Specific negative consequences are associated with shallow gamification. Gamification typically increases engagement—at least initially—but can lead to addiction, excessive competition, and off-task behavior [35]. Shallow gamification can even harm intrinsic motivation, especially when the gamification is perceived as controlling and when the topic was initially perceived as interesting [36].

E. Inefficient Learning

Learning is *inefficient* when the pace of learning is much lower than it could be. The difference between ineffective and inefficient learning is analogous to the difference between going in the wrong direction (ineffectivity) and going in the right direction but extremely slowly (inefficiency).

1) Causes: Three major sources of inefficiency are inappropriate difficulty (or level of guidance), inappropriate instructional method, and inappropriate feedback.

Cognitive load theory [37] explains how inappropriate difficulty leads to inefficient learning. Cognitive load has three components: (1) extraneous cognitive load, that is, effort unrelated to the learned topic, typically caused by the way the task is presented, (2) intrinsic cognitive load caused by the inherent complexity of the task, and (3) germane cognitive load, which is the effort caused by learning processes. Since the working memory is limited [38], [39], the extraneous and intrinsic load together can deplete it, not leaving any capacity for learning.

A crucial consequence is that minimally guided instruction is less efficient than more guided instruction unless the learners are sufficiently advanced to guide themself [31]. Without sufficient guidance and with all the mental effort focused on searching for a solution instead of learning, the students develop incomplete, disorganized, and sometimes even false knowledge.

That does not mean that we should strive for effortless learning. Germane cognitive load cannot emerge in a vacuum; it requires sufficient task complexity and associated intrinsic cognitive load. If the activity is so easy that the student does not need to exert effort, little learning will happen [40], [41]. Constructive activities (e.g., problem-solving) lead to better learning outcomes than passive engagement (e.g., watching a lecture) [42].

Inefficient learning can also be caused by an instructional method that is not suited for the given kind of knowledge. According to the Knowledge-Learning-Instruction framework [43], two key aspects are the complexity of the elicited learning process, which can range from simple (memory building) to complex (understanding), and the complexity of the knowledge, which can also range from simple (facts) to complex (principles). Instructional methods that elicit more complex learning processes are inefficient to teach less complex knowledge. For instance, self-explanation is an inefficient way to learn basic facts.

The third common source of inefficiency is inappropriate feedback [33]. The issue is not just missing feedback; equally unhelpful is feedback that is irrelevant, confusing, or not understandable [44].

2) Consequences: In addition to the obvious immediate consequences on students' competence, inefficient learning can also negatively impact motivation. If the students are aware of the inefficiency and attribute it to the system, it can cause distrust and possibly abandonment. If the students attribute the inefficiency to themself, it can cause frustration (i.e., the belief that they are not able to learn the topic).

F. Demotivation and Incompetence

The effects discussed so far are immediate mental processes caused by system actions. These affective and cognitive processes can lead to the two key negative, potentially long-term states: demotivation and incompetence. These two states are the basic reason why we care about undesirable system behaviors.

Demotivation refers to decreased intention to engage in a certain activity. Demotivation to use the system can be caused directly by distrust in the system or indirectly through demotivation to learn. Demotivation to learn is caused either by disinterest or frustration. Disinterested students do not want to learn the topic, while frustrated students do not believe that they are able to learn it.

Research on demotivation in classrooms is abundant [11], [12], [21], [23], [30], [45]. The results vary greatly depending on the specific context, but there are a few factors that are frequently reported as demotivating. All of them can be linked with the discussed negative affects: poorly organized materials, unfairness, and condescending or indifferent attitude cause distrust; monotonous learning activities, uninteresting materials, and irrelevant assignments cause disinterest; and unclear instructions, excessive difficulty, and frustrating failures cause frustration.

Incompetence—the other negative state—is a lack of durable and flexible knowledge. We use "knowledge" in a broad sense that includes memorized facts, nonverbalizable skills, understanding principles, and even meta-cognitive knowledge. Durability (long-term retention) is violated if the knowledge is soon forgotten; flexibility (transfer) is violated if the student is not able to apply the knowledge in new contexts.

Incompetence is the result of ineffective or inefficient learning—or no learning at all. Many authors do not use the term "competence" and instead specify the desired type of learning that leads to the competence as we have defined it (e.g., "long-term learning" [46] or "robust learning" [43]).

G. Nonuse, Misuse, and Poor Performance

Finally, the negative mental processes and states, which are not directly observable, lead to student behaviors that we can observe: nonuse, misuse, and poor performance.

When not being used, the learning environment does not contribute to learning, no matter how adaptive and intelligent it is. Nonuse does not have to be complete abandonment; partial nonuse is frequently the case. Nonuse can be mediated through other mental processes and states (e.g., distrust, demotivation), but it can also be a direct consequence of system actions. For

example, a student might not use a particular learning content because it does not render properly.

If the students are not motivated to use the learning environment but are forced to do so, they will misuse it. Misuse can take many forms, such as cheating [47], guessing [48], wheel spinning [49], excessive help-seeking [50], and other gaming-the-system behaviors [13]. Misuse is associated with negative affect [13] and worse learning [50].

Poor performance is an indicator of incompetence, but it can be as misleading as it is helpful. Performance is fluctuating and it is not a reliable way to assess competence during the learning or soon after [46]. In their excellent review of counterintuitive interactions between learning and performance, Soderstrom and Bjork warn that: "Conditions that appear to degrade acquisition performance are often the very conditions that yield the most durable and flexible learning." [46]

IV. Taxonomy of Stupidities: The Undesirable Behaviors of Learning Environments

Table I provides our proposal of the taxonomy of undesirable behaviors and a brief description of individual instances. In the discussion below, we focus on aspects common to all behaviors in each group.

A. Unusability

Usability is not specific to learning environments and thus does not get much attention in adaptive learning research. However, it is a key practical issue. As in other software systems, poor usability leads to distrust and abandonment of the learning environment. Several studies have highlighted the importance of usability in learning environments; for example, one of the recommendations for the development of adaptive learning environments is "Do not underestimate the importance of usability, flexibility, and scalability" [51]. Woolf [52] argues that measuring usability should be a part of the evaluation of learning environments.

Table I lists specific usability problems that can easily happen in a learning environment, such as response and navigation issues [8] and ignoring common vision deficiencies. Usability is best investigated through user studies, but specific, hidden unusability problems can also be detected from data. They can manifest as outliers in student activity; for example, if part of the question does not render properly, the consequence may be an unexpectedly low number of responses.

B. Presenting Deficient Content

Basic content errors, like grammatical mistakes and factual errors, are again not specific to learning environments. As in other settings, they lead to distrust in the system. In learning environments, however, such mistakes are even more important since they can also hamper learning by causing misconceptions and the illusion of knowledge.

Basic techniques for detecting deficient content are not specific to learning environments (spell-checkers, feedback from users). Specific methods may be useful for detecting missing content, for example, by finding discrepancies between instructional texts and interactive items, by comparison with a textbook, or by detecting missing recombinations of concepts.

A typical example of a hidden problem that can be easily detected in data is the *wrong answer*. Having incorrectly set which answer is correct often manifests as an outlier in difficulty. A more subtle situation is when the system does not accept valid alternatives in exercises with a constructed response (e.g., alternative translation in language learning or alternative way of writing a decimal number in mathematics). This problem can happen particularly easily for hand-written answers [8] or when the questions are created by students [51].

C. Insufficient Difficulty

Now, we get to undesirable behaviors that are specific to learning environments. The first one is an insufficient difficulty. Insufficient difficulty leads to boredom and disinterest but also to inefficient learning, as discussed in Section III. The issue of the proper choice of difficulty is complex. There is no clear boundary between "too easy" and "appropriate" difficulty, and there may be interactions with other elements like choice, novelty, and suspense [53].

As an undesirable behavior, we consider cases where it is evident that the task is too easy for the student and leads to boredom, disinterest, or learning inefficiency. Some items can be too easy for most students, for example, due to excessive scaffolding [54]. More frequently, an item would be too easy only for some students; the undesirability stems from ignoring evidence of prior knowledge (e.g., recommending basic addition to a student who has already solved exercise on logarithms). Another common problem is repetitiveness—due to a low number of available items, the system presents students repeatedly with the same items, which makes them too easy and boring. A specific type of repetitiveness is caused by the failure of the system to recognize student mastery—for example, because of a wrong parametrization of a mastery criterion [55]—which leads to overpractice.

Insufficient difficulty can be detected, for example, by difficulty analysis (too high success rate of items) and by analysis of student activity (repeated answers for the same item in a short time).

D. Undesirable Difficulty

An opposite problem to insufficient difficulty is undesirable difficulty—cases where the difficulty of learning materials is too high. Undesirable difficulty leads to frustration and inefficient learning, as discussed in Section III.

As already mentioned, the issue of appropriate difficulty is complex. There are many desirable difficulties that lead to better long-term learning [40]. We again consider cases where the difficulties are clearly undesirable, for example, when the goal is unclear or when the task requires many tedious and monotonous steps to complete. Table I gives other examples.

Whether a difficulty is appropriate is often connected to the type of cognitive load [56] (intrinsic, extraneous, germane). Extraneous complexity is quite a clear source of undesirable difficulty, but excessive intrinsic cognitive load is more subtle. It interacts with limits of working memory, and these are

TABLE I
TAXONOMY OF STUPIDITIES: THE UNDESIRABLE BEHAVIORS OF LEARNING ENVIRONMENTS

Unusability	
no response	Not responding to a student's request (e.g., a chosen task does not display due to a technical bug).
slow response	Responding too slowly to student's actions.
confusing interface	Providing a user interface that is confusing, so that it is not clear how to perform actions towards a goal, or the actions are nonintuitive.
superfluous actions	Requiring redundant or repetitive actions in order to learn (e.g., an additional click after each item to display feedback).
exclusion	Assigning a task that the student cannot solve due to a disability (e.g., vision deficiency) or a technical limitation (e.g., working on a tablet without a mouse).
poor organization	Presenting incoherent organization of learning materials so that some topics are difficult to find.
Presenting deficient content	
text mistake	Displaying content with grammatical mistakes and misspelled words.
factual error	Displaying wrong information in learning materials.
wrong answer	Showing a wrong answer as correct or not accepting a valid correct answer (e.g., an alternative formulation).
content mismatch	Showing a wrong learning resource (e.g., an explanation for another item).
missing content	Omitting critical parts of some topic.
inferior texts	Displaying poorly written learning materials or questions (e.g., vague, unclear, incoherent, or dull).
inferior media	Presenting poor-quality or highly inconsistent images, audio, video, or simulations.
Insufficient difficulty	
insufficient complexity	Assigning a task that is too easy due to its internal structure or excessive scaffolding.
ignoring knowledge	Assigning a task that is too easy due to prior knowledge (e.g., recommending basic addition to an adult).
repetitiveness	Repeatedly presenting the same or very similar items (e.g., because there are few items in a problem set).
unrecognized mastery	Forcing the student to solve too many items on the same topic (overpractice).
Undesirable difficulty	
extraneous complexity	Presenting a task in such a way that it generates unnecessary extraneous cognitive load.
excessive complexity	Assigning a task that is too difficult due to its internal structure.
ignoring prerequisites	Assigning a task that the student cannot solve due to missing prerequisites or insufficient fluency in prerequisites.
excessive laboriousness	Assigning a tedious task that requires many trivial and monotonous steps to complete.
steep curve	Presenting items in a sequence where the difficulty is increasing too quickly.
unclear goal	Presenting tasks in such a way that the goal not clear, so the student does not know how to proceed.
harsh assessment	Using too harsh evaluation criteria (e.g., no credit for multi-step item with just one mistake), which leads students to perceive their performance as insufficient even when the given task was of suitable difficulty.
Ineffective teaching strategy	
misdirected gamification	Rewarding irrelevant activities (e.g., solving too easy tasks) or encouraging excessive off-task behavior (e.g., tuning avatar appearance).
premature mastery declaration	Declaring mastery too early and proposing students to solve more complex topics (underpractice).
shallow teaching	Teaching only recall and recognition, not integration and higher-level skills.
massed instruction	Promoting cramming and not supporting spaced repetition and interleaving of topics.
inappropriate recommendations	Recommending activities that are clearly inappropriate for a particular student.
noninteractivity	Presenting students information without giving them an opportunity to practice.
Inappropriate feedback	
misleading feedback	Giving feedback that reinforces misconceptions (e.g., rewarding students for a low-quality solution or "correct answer for the wrong reason").
missing feedback	Not giving feedback on performance.
unhelpful feedback	Giving feedback that is inappropriate for the target audience or performed mistake.

related to fluency in prerequisite knowledge. So the presented types of undesirable difficulties can overlap.

The main tool to detect these undesirable behaviors is the analysis of item difficulty (e.g., visualization of difficulty distribution and detection of outliers).

E. Ineffective Teaching Strategy

By teaching strategy (also called instructional method), we denote the approach that the learning environment uses to select topics and tasks that are presented to students and the specific form of their presentation. Some teaching strategies are ineffective (i.e., they cause students to spend time on inappropriate activities). As opposed to previous cases, students may feel good and have a positive impression about their learning, but their knowledge is incomplete, shallow, or fragile, and the students develop an illusion of knowledge.

One source of this illusion can be gamification, which can lead to unforeseen and undesirable effects. For example, it can motivate students to maximize some type of activity—such as collecting coins by solving easy exercises—which is not in the interest of their learning. Possible undesirable effects of gamification include addiction, excessive competition, and off-task behavior [35].

Underpractice is a problem when a system declares mastery too early [55] and students move to more advanced topics before they have sufficiently mastered the current one. Underpractice can often manifest itself later as undesirable difficulty, but we want to address the root cause, which is the premature mastery declaration.

Shallow teaching, massed instruction, and noninteractivity involve ignoring desirable difficulties [40] that are known to improve learning in the long term, such as spaced repetition and interleaving. They differ from the other undesirable behaviors in the *insufficient difficulty* category in the way they are perceived by students—the ineffective teaching strategies typically do not cause negative affect and they do lead to the illusion of knowledge. Employing more effective teaching strategies involves nontrivial trade-offs—materials that support deeper and more active learning may be costly to develop. As an undesirable behavior, we consider cases where the learning environment already uses the effective strategies for some topics, yet there is a topic that is taught ineffectively, and it would be relatively easy to change (e.g., to create items to practice the topic).

This type of undesirable behavior can be detected mostly from student activity data—particularly analyzing the portion of time spent in different types of activities.

F. Inappropriate Feedback

Feedback is a key aspect of learning and poor feedback degrades the efficiency of learning [44].

The basic form of feedback is the information about the correctness of answers. Even when this feedback is factually correct, it can be misleading. If a student gives a correct answer for a wrong reason and the system provides only a confirmation of correctness, the student may fixate a misconception. As a specific case, consider a real situation from our experience: a comparison of fractions. Suppose that we create a set of easy examples as a warm-up exercise and we create this set mechanically by choosing top N examples with the highest success rate. We may end up with examples where the ordering of the two fractions is the same as the ordering of their nominators—which is a common but wrong belief. Letting students practice on such a set would just strengthen this wrong misconception. Such cases are tricky to detect automatically; they call for a more complex analysis with the use of student modeling techniques.

Another common type of feedback is the use of explanations after answers. In this case, the basic type of undesirable behavior is simply "missing feedback" (i.e., a missed opportunity). Using data on student activity, it is easy to detect cases where students often answer incorrectly and feedback is missing. These cases are natural candidates for the extension of content.

More complicated to detect is "unhelpful feedback"—cases where feedback message is available but inappropriate; for example, it is too complex for the target audience, does not clearly explain the reasoning behind the correct answer, or does not address the reason for typical mistakes. Detection of this type of problem requires a more detailed analysis of answers and feedback content.

V. Detecting Defects and Undesirable Behaviors

Our focus is on hidden defects that can be detected using data-driven techniques. In this section, we provide an overview of such techniques. We discuss mainly the general ideas behind them: what input they take, what do they try to achieve, what output they provide.

We sort techniques based on common concepts they utilize (complexity, similarity, difficulty, student modeling). Fig. 2 provides an overview of used data, discussed techniques, and their relations. These techniques were typically developed for other purposes but often can be naturally employed for the detection of defects.

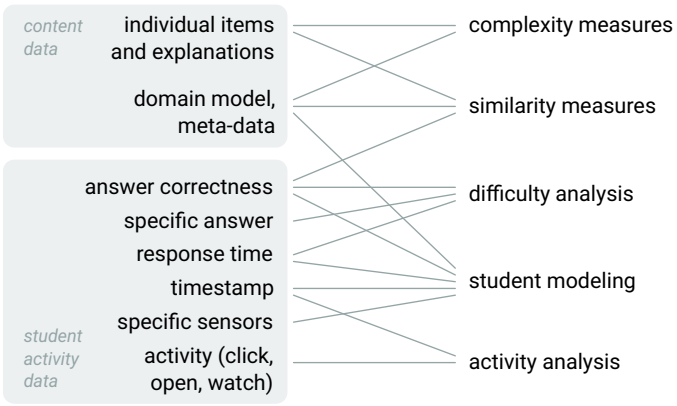

Fig. 2. Overview of data sources used by different types of techniques.

Fig. 3 illustrates several possible visualizations of the results of data analysis techniques. The graphs are based on real experiences with the development of an adaptive learning environment Umíme (`umimeto.org`), which is used by 10 % Czech schools and contains a thousand knowledge

components and a hundred thousand items. The graphs do not, however, depict real data. For presentation clarity, the data and presentation are simplified to illustrate multiple techniques and issues in a compact space. In the discussion below, we provide more details on specific examples from our experience and also references to literature.

A. Complexity Measures

Complexity is an intrinsic item characteristic, which aggregates item aspects that influence how students solve the item; complexity measure is specific quantification of complexity [57]. Complexity measures are computed based only on content data; they do not use data on student performance.

Typical examples of complexity measures are readability measures for text [58], which are relevant for many educational domains. For detecting defects in educational items, it may be useful to use age-of-acquisition ratings of words [59] to detect the usage of vocabulary inappropriate for a target audience. More specialized examples of complexity measures are the number of distinct operators used in math word problems, the cyclomatic complexity of a sample solution for a programming exercise, and the properties of state space of a logic puzzle [57].

1) Undesirable Difficulty: Complexity measures are relevant mostly for detecting undesirable difficulty, particularly the excessive and extraneous complexity issues. The basic approach is to use outlier detection with respect to complexity measures. A typical example is the detection of items with significantly more complex text (as measured by readability formulas) than the rest of a problem set. For this analysis, a variation of Fig. 3 A is useful. When the boxplots show values of a readability measure for sets of texts, we can visually detect texts which are too complex.

We can measure complexity not just for individual items but also for whole sets of items and use this measurement to find mistakes in the domain model; for example, complexity measures are expected to be correlated with assigned grades (for how old children the topic is intended), so outliers in such a correlation are worth verification.

2) Insufficient Difficulty: The detection of outliers is also relevant for the detection of insufficient difficulty. In addition to the use of generic complexity measures, it is useful to develop specific detectors of unintended low complexity—aspects of learning materials that allow students to answer correctly even without a proper understanding of the topic. For example, in multiple-choice questions, it can happen that a simple heuristic ("select the option with the longest text" or "select the last option") has a high success rate; that is, students can successfully answer items without real understanding, which is inappropriate.

B. Similarity Measures and Projections

Item similarity is a property of a pair of items. A similarity measure quantifies the similarity of two items [60]. A simple similarity measure for text items is Levenshtein edit distance or Jaccard similarity over bag-of-words representation; more complex measures take the structure of items into account. It is also possible to use embeddings (like word2vec) [61]. Item similarity can also be computed based on student performance (e.g., as a correlation of student answers on two items [60]).

To detect defects, we can use similarity measures together with outlier detection techniques or use them to create projections (e.g., PCA, tSNE) and check them visually.

1) Deficient Content: Fig. 3 C provides an example of the potential use of similarity measures. The example corresponds to the case where we have alternate choice items (e.g., the choice of correct determiner in the practice of English grammar), items with the same answer have the same color. The visualization shows several outliers, which are potential mistakes or unsuitable items. It also shows a gap between two groups of items; this may indicate missing content and provide inspiration of content authors (together with inspection of specific items).

Similarity measures can also be used for the detection of content mismatch. Similarity can be quantified using different data, and we expect that different ways to quantify similarity are in approximate agreement. For example, we have used this approach to detect cases of wrongly specified feedback messages (e.g., two items had swapped messages).

2) Insufficient and Undesirable Difficulty: Problems with insufficient and undesirable difficulty are straightforwardly detected using difficulty analysis. Similarity measures can, in some cases, provide more specific insight into these problems and suggest potential solutions. For example, one source of insufficient difficulty is repetitiveness caused by items that are too similar or even complete duplicates. Such a situation can happen easily when the content for some topic is prepared by multiple authors. Similarity measures can detect these cases.

The inverse problem is too large heterogeneity, which often leads to undesirable difficulty (particularly steep curve). In the similarity analysis, this problem manifests itself by significant gaps in the projection based on similarity measures.

C. Difficulty Analysis

Item difficulty, as opposed to the above-discussed complexity, is based on the observed performance of students [57]. Basic difficulty measures are the failure rate (the ratio of students who answer incorrectly) and median response time. These basic measures have disadvantages, particularly susceptibility to biases in data (e.g., attrition bias and self-selection bias [62]). These disadvantages can be (partially) mitigated by using difficulty measures derived from student modeling techniques (discussed below). However, for the purposes of detection of undesirable behaviors, even the basic measures are often sufficient.

1) Undesirable and Insufficient Difficulty: One way to detect undesirable and insufficient difficulty is to combine difficulty and complexity measures, as illustrated in Fig. 3 B. We can also use a similar type of analysis with two difficulty measures (e.g., success rate and median time). The outliers can be either detected visually in such scatter plots or by using standard outlier detection techniques [63]. To provide a specific example, consider a knowledge component "transforming fractions to percents." The analysis of outliers

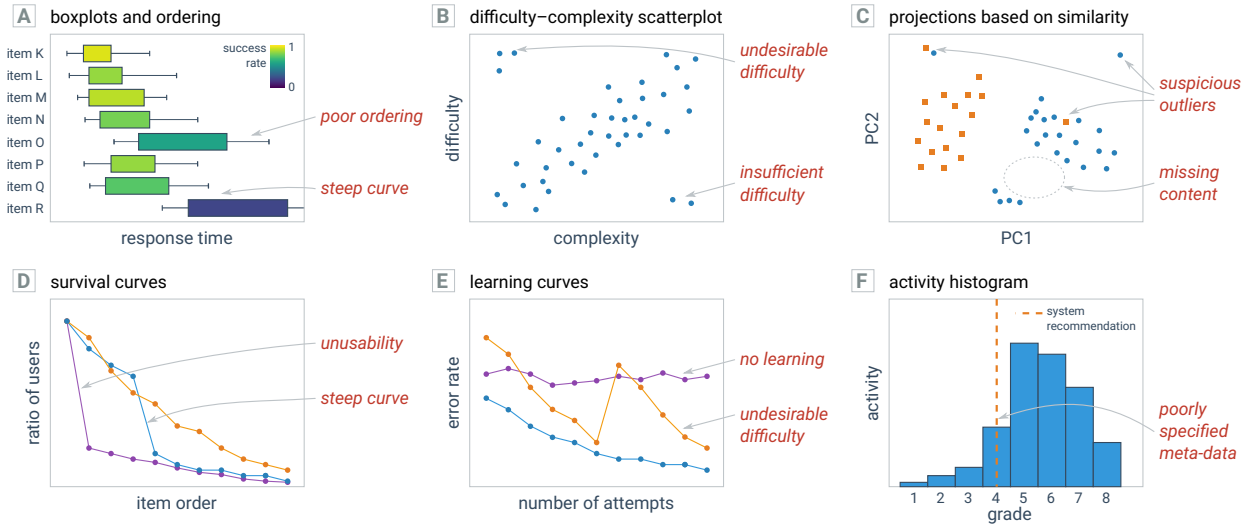

Fig. 3. Examples of data analysis visualizations and undesirable behaviors that they can help to uncover.

detected items $\frac{24}{30}, \frac{6}{8}$ in a set where typical items are like $\frac{3}{4}, \frac{3}{100}$. Such situations can be solved in different ways. The basic solution is to remove the outlying items. But we can also add more items of a similar type and create a new problem set which can be preceded by practice of fluency on simplifying fractions.

Fig. 3 A provides another type of visualization of difficulty, which can be used to detect steep difficulty curves and poor ordering of items. Boxplots show the distribution of response times; the color corresponds to the success rate. The items are visualized in the ordering used in the learning environment. The graph shows that that the difficulty increases mostly gradually, but there is one poorly ranked item (O), and the last item (R) is too difficult.

2) Deficient Content: Difficulty analysis can also be used for the detection of deficient content, particularly wrong answers. One simple approach is the heuristic "the success rate is zero," which can detect clear typos and major technical bugs in item representation. A slightly more nuanced approach uses not just the correctness of answers but also their specific values. A simple, natural heuristic is "the most common unaccepted answer is more common than the denoted correct answer." Of course, not all cases that these heuristics returns are wrong answers. It may also be a case of undesirable difficulty or even desirable difficulty question, which contains an explanation that forces students to correct their misconception. Even in these other cases, however, it is worthwhile to pay attention to a given item and reconsider its formulation.

3) Ineffective Teaching Strategy: We can compute difficulty measures not just for individual items but also for whole item sets (the practice of a specific skill), for example, using metrics like the ratio of students who reach mastery or median time to reach mastery. Too low or high values are often indicative of a problem. There can be multiple reasons for these values, and additional analysis is typically necessary to determine the specific reason. Too low difficulty of achieving mastery can be caused by premature mastery declaration (poorly set mastery criterion), misdirected gamification (students are practicing simple skills just to get badges), or massed instruction. Too high difficulty is often caused simply by the undesirable difficulty of individual items. It can, however, also be an indirect result of premature mastery in the practice of prerequisite skills.

4) Inappropriate Feedback: To detect misleading or unhelpful feedback, we may perform a differential analysis of difficulty—computing the difference in the success rate of different groups of students, depending on whether they have already seen a specific feedback message. If there is no change in success rate after seeing feedback, the used feedback is probably not effective. This type of analysis, however, is nontrivial due to attrition and selection biases. To perform this analysis properly, it is useful to employ student modeling techniques, which we cover next.

D. Student Modeling

The main purpose of student modeling [64] is typically to obtain an estimate of a student state, which is used to guide personalized behavior or to provide feedback to students. Student modeling mostly focuses on the estimation of cognitive state, but there are also techniques for estimating affective or meta-cognitive state (e.g., engagement, boredom, confusion, or frustration [65]). These techniques may be useful for the detection of unusability. Student modeling can also be used to detect undesirable student behavior like gaming-the-system [13], guessing [48], excessive help-seeking [8], or cheating [47], [66]. These student behaviors are sometimes caused by undesirable behaviors of a learning environment.

In the following, we focus on modeling of cognitive state. The primary purpose of student models is to model students' states, not to detect undesirable behaviors of learning environments. Nevertheless, they can be useful in a variety of ways. We outline several directions; we believe that there is further potential in this direction.

1) Deficient Content: Detection of wrong answers can utilize the concept of item discrimination parameter, which

is used, for example, in item response theory [67]. Item discrimination specifies how well an item discriminates between students with different skills. If the item discrimination is negative, it means that students with high skills have a lower chance of answering correctly. This is not expected; such a result is often indicative of a wrongly specified item.

2) Unusability: Some student modeling approaches utilize clustering of students; these models may, as a side effect, detect exclusion problems. A specific example is provided by the mixture modeling of skill performed in the context of DuoLingo [68]. Using this approach, the authors identified a set of students who systematically skipped items with sound. Based on this, they conjectured that these students are either hearing-impaired or do not have working speakers (i.e., that the learning environment exhibits some kind of exclusion).

3) Undesirable and Insufficient Difficulty: A student model may be able to estimate skill and thus detect situations where a student is practicing something that he already knows (insufficient difficulty; ignoring knowledge). Student modeling is intertwined with domain modeling and can be used to detect cases of undesirable difficulty due to a poor domain model, for example, failure to distinguish different knowledge components or absence of important prerequisite relations. The mapping of items to knowledge components is typically denoted Q-matrix [69] and used in models like the additive factors model [70]. A general approach to spot problems in the Q-matrix is the use of learning curves [71]—searching for knowledge components with flat or increasing error curves. There are also specific techniques for Q-matrix refinement [72], and some works have a specific focus on prerequisite analysis and discovery [73], [74].

4) Ineffective Teaching Strategy: Student models may also be used to detect problems with mastery criteria. In the case of premature mastery declaration, students may have difficulties with solving follow-up topics. In the case of unrecognized mastery, students are forced to answer items that are easy for them. Note that student models are often used as a basis for mastery criteria, and thus a poorly chosen or fitted student model may be the source of a problem. At the same time, another student model may be useful for detecting the problem.

5) Inappropriate Feedback: Whenever a learning environment uses a student model to guide its behavior, a poor model fit is indicative of a potential problem. A specific example is described by Brinkhuis *et al.* [75]. They detected a case where estimates of student skill underwent sudden jumps, which was unexpected for the used modeling approach (Elo rating system). This led to the identification of misleading feedback. For a specific knowledge component, the used algorithm (based on dynamic estimation of students' skills and items' difficulties) led to reinforcing misconceptions—erroneous local strategies that appeared to work well for many successive items. The solution was to make the algorithm less adaptive (for a particular problem set).

Learning curves [71] are a tool that is used to evaluate student modeling techniques. In its basic form, it shows the student error rate depending on the number of practice opportunities. Fig. 3 E shows an illustrative example. The blue curve shows a desirable shape of the curve. If the curve is flat (as the violet one), it suggests that students are not learning, which may be, for example, due to inappropriate feedback. However, this type of analysis is nontrivial due to student attrition and ordering biases [76].

E. Activity Analysis

Finally, we discuss techniques that focus on the distribution of student activity within the learning environment and on interaction patterns between students and items.

1) Unusability: Unusability problems, for example, no response, slow response, exclusion, or poor organization, can be often detected using just counts of visits (e.g., how many times in a day was an item answered) and their monitoring using simple heuristics (e.g., listing of items with zero counts) and outlier detection techniques (e.g., outliers within a set, temporal outliers).

2) Deficient Content: Simple popularity statistics can also be useful for detecting missed opportunities—missing content that is easy to add and is useful. Such opportunities can be detected by listing popular topics, for which there are fewer practice opportunities than for other, less popular ones.

Other deficient content can be detected using analysis of timing information—inferior text or media may manifest themselves as outliers with respect to response times.

3) Insufficient and Undesirable Difficulty: Another application of visit counts is the detection of item repetitiveness, which is caused by an insufficient number of practice items for a particular topic. This undesirable behavior can be detected by analysis of activity, for example, by computing "how many students solve all items in a knowledge component without reaching mastery" or "how many students repeatedly solve some item within a single session." In our experience, the simple total number of students who encounter repetitiveness is a very useful tool for prioritization for content creators: knowledge components with higher numbers deserve a higher priority.

The undesirable difficulty often manifests itself in survival curves, which show the proportion of students that reach ("survive until") a given item. [77], [78]. Fig. 3 D shows an illustrative example—the orange line is a typical curve for a sequence of items with a fixed progression. The blue curve shows a large drop after the 4th item, which suggests that this item is problematic. The violet line shows a significant drop after the first item, which may mean high difficulty, but more probably some more fundamental usability problem.

4) Poor Teaching Strategy: The activity analysis can be connected with data about students to make it more informative. For example, consider the information about student grade (year). Using this information, for each knowledge component, we can decompose student activity by grades, visualize it as a histogram, and compare it to the mapping to grade in the used domain model. A mismatch between the real student activity and a supposed student activity (as specified in the domain model) is typically indicative of problems with teaching strategy, which can manifest as inappropriate recommendations. Fig. 3 F illustrates an example of this type

of analysis: for a specific topic, it shows a histogram of student activity by grades, together with the grade assigned to the topic in the domain model. The student activity data suggest that the assigned grade should be larger.

5) Inappropriate Feedback: Analysis of timing information can detect problems with feedback messages—to do so, we need to measure the time between the display of the feedback message and the following student action (e.g., continuing to next item). If this time is very low, it means that the message was not read by a student. A high proportion of such cases means that feedback messages are not working as intended.

VI. Conclusion

Finally, we provide a summary of our perspective, the proposed taxonomy of undesirable behaviors and its relation to data-driven techniques for finding them and highlight the main implications for practice and research.

A. Avoiding Stupidity Perspective

We advocate a shift of focus in the development of adaptive learning environments. Currently, research and development often aim at "optimization" of the learning process. However, learning is complex. It is not very useful to optimize one aspect of the process (e.g., the optimal gap for spaced repetition) when other aspects are realized poorly (e.g., feedback is misleading). Moreover, learning involves trade-offs and complex interactions between desirables. Optimization of any single aspect can be potentially dangerous, especially for aspects that are realistic to measure. Optimizing engagement (measured by spent time) can lead to shallow gamification, "fun without learning," and addiction. Optimizing performance at the end of the session can lead to massed practice. Optimizing performance after delay (within the learning environment) can indirectly cause the algorithm to avoid difficult content (e.g., higher-level skills). Similarly, unintended consequences can follow from the optimization of student autonomy or social interactions.

We believe that instead of aiming at optimality, we should focus on avoiding stupidity. A learning environment can be wrong in many ways. Once we manage to avoid being wrong, we should end up with a rather good, efficient learning environment.

B. Applying the Perspective

To make the avoiding stupidity perspective actionable, we propose a taxonomy of specific undesirable behaviors and an overview of detectors that are immediately applicable in the development of learning environments. Fig. 4 provides a coarse mapping of undesirable behaviors and techniques. The figure gives a simplified, high-level summary of the discussion of applications provided in Section V, that is, it corresponds to techniques that we are aware of and that we have mentioned. It certainly could (and should) be subject to revisions in future work.

The figure illustrates that the described techniques are, to a large degree, complementary—each of them is suitable for detecting different types of undesirable behaviors. Even in cases where multiple techniques can be used to detect the same type of problem, there is often a complementarity aspect—different techniques may detect different specific occurrences.

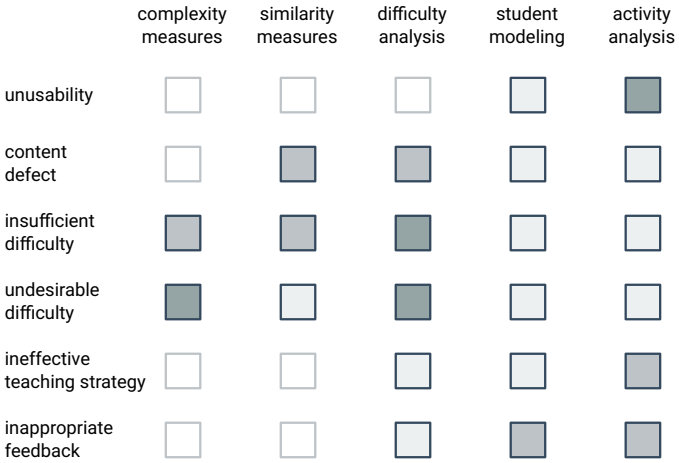

Fig. 4. Mapping of undesirable behaviors and techniques to uncover them; darker color means that a given technique is more relevant to the detection of a given type.

Our experience is that once we explicitly describe specific undesirable behavior, it is often possible within a few hours to build a detector, find a nontrivial list of its occurrences, uncover and correct specific defects, and thus improve the learning environment. The avoiding stupidity perspective becomes even more powerful once it is incorporated into the design process; that is, when the design of the environment is done in such a way that detection of undesirable behaviors and iterative improvement are facilitated (e.g., by collecting suitable data and implementing support for easy revision of existing items).

C. Researching the Perspective

To further develop the perspective of avoiding stupidity, more research is needed. It would be beneficial to study in more detail the nuances of relations between system actions and student reactions depicted in Fig. 1. For further development of defect detectors, it would be useful to perform a quantitative evaluation of their performance. This is not easy. Typically, once a defect is detected, it is immediately removed (that is, after all, the point of detecting it). In order to build a dataset suitable for the evaluation of defect detectors, it would be necessary to systematically collect their occurrences. Note that such benchmark sets of "bugs" are commonly used in software engineering in the development of automatic testing and verification techniques [79], [80]. Another possibility is to take a high-quality data set, create artificial defects (e.g., by modifying some items and their answers) and then evaluate the ability of techniques to find them.

A related issue is prioritization. On which undesirable behaviors should we focus? Which are the most important (severe, frequent)? This issue will often depend on the specific circumstances of a particular learning environment. Nevertheless, it may be a worthwhile research direction to provide some

general techniques for prioritization; Mian *et al.* [8] provide a step in this direction.

The perspective of avoiding stupidity also provides interesting impulses for novel research directions. The most natural one is the development of novel data-driven defect detectors for specific cases of problems. It may also be useful to reinterpret or modify existing techniques for the purpose of defect detection. We briefly mentioned several examples of this type in the case of student modeling techniques. There are, surely, many other opportunities.

REFERENCES

[1] X. Wang, C. Rose, and K. Koedinger, "Seeing beyond expert blind spots: Online learning design for scale and quality," in *Proc. Human Factors in Computing Systems*, 2021, pp. 1–14.

[2] Software & Systems Engineering Standards Committee, "IEEE standard classification for software anomalies," IEEE Computer Society, Tech. Rep., 2009.

[3] V. Aleven, E. A. McLaughlin, R. A. Glenn, and K. R. Koedinger, *Handbook of Research on Learning and Instruction*. Routledge, 2016, ch. Instruction based on adaptive learning technologies, pp. 522–559.

[4] R. Liu and K. R. Koedinger, "Closing the loop: Automated data-driven cognitive model discoveries lead to improved instruction and learning gains." *J. of Educ. Data Mining*, vol. 9, no. 1, pp. 25–41, 2017.

[5] H. Hassani, E. S. Silva, S. Unger, M. TajMazinani, and S. Mac Feely, "Artificial intelligence (ai) or intelligence augmentation (ia): What is the future?" *AI*, vol. 1, no. 2, pp. 143–155, 2020.

[6] R. S. Baker, "Stupid tutoring systems, intelligent humans," *Int. J. of Artif. Intell. in Educ.*, vol. 26, no. 2, pp. 600–614, 2016.

[7] S. M. McNee, N. Kapoor, and J. A. Konstan, "Don't look stupid: avoiding pitfalls when recommending research papers," in *Proc. Computer supported cooperative work*, 2006, pp. 171–180.

[8] S. Mian, M. Goswami, and J. Mostow, "What's most broken? design and evaluation of a tool to guide improvement of an intelligent tutor," in *Proc. Artificial Intelligence in Education*, 2019, pp. 283–295.

[9] G. J. Myers, T. Badgett, T. M. Thomas, and C. Sandler, *The art of software testing*. Wiley Online Library, 2004, vol. 2.

[10] Y. D. Wang, "Building student trust in online learning environments," *Dist. Educ.*, vol. 35, no. 3, pp. 345–359, 2014.

[11] J. Gorham and D. M. Christophel, "Students' perceptions of teacher behaviors as motivating and demotivating factors in college classes," *Commun. Quarterly*, vol. 40, no. 3, pp. 239–252, 1992.

[12] Q. Zhang, "Teacher misbehaviors as learning demotivators in college classrooms: A cross-cultural investigation in china, germany, japan, and the united states," *Commun. Educ.*, vol. 56, no. 2, pp. 209–227, 2007.

[13] R. Baker, J. Walonoski, N. Heffernan, I. Roll, A. Corbett, and K. Koedinger, "Why students engage in "gaming the system" behavior in interactive learning environments," *J. Interact. Learning Res.*, vol. 19, no. 2, pp. 185–224, 2008.

[14] S. M. McNee, J. Riedl, and J. A. Konstan, "Making recommendations better: an analytic model for human-recommender interaction," in *Proc. Human Factors in Computing Systems*, 2006, pp. 1103–1108.

[15] J. A. Konstan and J. Riedl, "Recommender systems: from algorithms to user experience," *User Model. and User-Adapted Interact.*, vol. 22, no. 1, pp. 101–123, 2012.

[16] S. Hidi and J. M. Harackiewicz, "Motivating the academically unmotivated: A critical issue for the 21st century," *Rev. of Educ. Res.*, vol. 70, no. 2, pp. 151–179, 2000.

[17] U. Schiefele, "Interest and learning from text," *Sci. Stud. of Reading*, vol. 3, no. 3, pp. 257–279, 1999.

[18] S. Hidi and K. A. Renninger, "The four-phase model of interest development," *Educ. Psychol.*, vol. 41, no. 2, pp. 111–127, 2006.

[19] G. Schraw, T. Flowerday, and S. Lehman, "Increasing situational interest in the classroom," *Educ. Psycholog. Rev.*, vol. 13, no. 3, pp. 211–224, 2001.

[20] S. Hidi and W. Baird, "Interestingness—a neglected variable in discourse processing," *Cogn. Sci.*, vol. 10, no. 2, pp. 179–194, 1986.

[21] J. Falout, J. Elwood, and M. Hood, "Demotivation: Affective states and learning outcomes," *System*, vol. 37, no. 3, pp. 403–417, 2009.

[22] K. Kikuchi, "Listening to our learners' voices: what demotivates japanese high school students?" *Lang. Teach. Res.*, vol. 13, no. 4, pp. 453–471, 2009.

[23] H. Sakai and K. Kikuchi, "An analysis of demotivators in the efl classroom," *System*, vol. 37, no. 1, pp. 57–69, 2009.

[24] M. Csikszentmihalyi, *Flow: The psychology of optimal experience*. Harper & Row New York, 1990, vol. 1990.

[25] A. Lindstedt, A. Koskinen, J. McMullen, M. Ninaus, and K. Kiili, "Flow experience and situational interest in an adaptive math game," in *Proc. Games and Learning Alliance*, 2020, pp. 221–231.

[26] E. M. Anderman and C. A. Wolters, *Handbook of Educational Psychology*. Lawrence Erlbaum Associates Publishers, 2006, ch. Goals, Values, and Affect: Influences on Student Motivation, pp. 369–389.

[27] T. W. Malone, *Aptitude, Learning and Instruction III: Conative and affective process analysis*. Erlbaum, 1987, ch. Making learning fun: A taxonomic model of intrinsic motivations for learning.

[28] A. M. Toda, A. C. Klock, W. Oliveira, P. T. Palomino, L. Rodrigues, L. Shi, I. Bittencourt, I. Gasparini, S. Isotani, and A. I. Cristea, "Analysing gamification elements in educational environments using an existing gamification taxonomy," *Smart Learn. Environ.*, vol. 6, no. 1, pp. 1–14, 2019.

[29] M. Kapur, "Examining productive failure, productive success, unproductive failure, and unproductive success in learning," *Educ. Psychol.*, vol. 51, no. 2, pp. 289–299, 2016.

[30] G. Chambers, "Taking the 'de'out of demotivation," *Lang. Learn. J.*, vol. 7, no. 1, pp. 13–16, 1993.

[31] P. A. Kirschner, J. Sweller, R. E. Clark, P. Kirschner, and R. Clark, "Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching," *Educ. Psychol.*, vol. 41, no. 2, pp. 75–86, 2006.

[32] M. V. Covington, "The self-worth theory of achievement motivation: Findings and implications," *The Element. School J.*, vol. 85, no. 1, pp. 5–20, 1984.

[33] A. C. Butler and H. L. Roediger, "Feedback enhances the positive effects and reduces the negative effects of multiple-choice testing," *Memory & Cogn.*, vol. 36, no. 3, pp. 604–616, 2008.

[34] M. Schwartz, "Khan academy: The illusion of understanding," *Online Learn. J.*, vol. 17, no. 4, 2013.

[35] F. R. Andrade, R. Mizoguchi, and S. Isotani, "The bright and dark sides of gamification," in *Proc. Intelligent Tutoring Systems*, 2016, pp. 176–186.

[36] M. D. Hanus and J. Fox, "Assessing the effects of gamification in the classroom: A longitudinal study on intrinsic motivation, social comparison, satisfaction, effort, and academic performance," *Comput. & Educ.*, vol. 80, pp. 152–161, 2015.

[37] F. Paas, A. Renkl, and J. Sweller, "Cognitive load theory and instructional design: Recent developments," *Educ. Psychol.*, vol. 38, no. 1, pp. 1–4, 2003.

[38] G. A. Miller, "The magical number seven, plus or minus two: Some limits on our capacity for processing information." *Psychol. Rev.*, vol. 63, no. 2, p. 81, 1956.

[39] N. Cowan, "The magical mystery four: How is working memory capacity limited, and why?" *Curr. Direct. in Psychol. Sci.*, vol. 19, no. 1, pp. 51–57, 2010.

[40] E. L. Bjork and R. A. Bjork, *Psychology and the real world: Essays illustrating fundamental contributions to society*. Worth Publishers, 2011, no. 59-68, ch. Making things hard on yourself, but in a good way: Creating desirable difficulties to enhance learning.

[41] H. L. Roediger III and M. A. Pyc, "Inexpensive techniques to improve education: Applying cognitive psychology to enhance educational practice," *J. of Appl. Res. in Memory and Cogn.*, vol. 1, no. 4, pp. 242–248, 2012.

[42] M. T. Chi and R. Wylie, "The icap framework: Linking cognitive engagement to active learning outcomes," *Educ. Psychol.*, vol. 49, no. 4, pp. 219–243, 2014.

[43] K. R. Koedinger, A. T. Corbett, and C. Perfetti, "The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning," *Cogn. Sci.*, vol. 36, no. 5, pp. 757–798, 2012.

[44] J. Hattie and M. Gan, *Handbook of Research on Learning and Instruction*. Routledge, 2011, ch. Instruction based on feedback, pp. 249–271.

[45] Y. B. Khouya, "Students demotivating factors in the efl classroom: The case of morocco." *Advances in Lang. and Liter. Stud.*, vol. 9, no. 2, pp. 150–159, 2018.

[46] N. C. Soderstrom and R. A. Bjork, "Learning versus performance: An integrative review," *Perspect. on Psychol. Sci.*, vol. 10, no. 2, pp. 176–199, 2015.

[47] H. Corrigan-Gibbs, N. Gupta, C. Northcutt, E. Cutrell, and W. Thies, "Deterring cheating in online environments," *ACM Trans. on Comput.-Human Interact.*, vol. 22, no. 6, pp. 1–23, 2015.

[48] R. S. d Baker, A. T. Corbett, and V. Aleven, "More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing," in *Proc. Intelligent Tutoring Systems*, 2008, pp. 406–415.

[49] J. E. Beck and Y. Gong, "Wheel-spinning: Students who fail to master a skill," in *International conference on artificial intelligence in education*, 2013, pp. 431–440.

[50] V. Aleven, I. Roll, B. M. McLaren, and K. R. Koedinger, "Help helps, but only so much: Research on help seeking with intelligent tutoring systems," *Int. J. of Artif. Intell. in Educ.*, vol. 26, no. 1, pp. 205–223, 2016.

[51] H. Khosravi, S. Sadiq, and D. Gasevic, "Development and adoption of an adaptive learning system: reflections and lessons learned," in *Proc. ACM Technical Symposium on Computer Science Education*, 2020, pp. 58–64.

[52] B. P. Woolf, *Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning*. Morgan Kaufmann, 2010.

[53] J. D. Lomas, K. Koedinger, N. Patel, S. Shodhan, N. Poonwala, and J. L. Forlizzi, "Is difficulty overrated? the effects of choice, novelty and suspense on intrinsic motivation in educational games," in *Proc. Human Factors in Computing Systems*, 2017, pp. 1028–1039.

[54] J. Jennings and K. Muldner, "When does scaffolding provide too much assistance? a code-tracing tutor investigation," *Int. J. of Artif. Intell. in Educ.*, pp. 1–36, 2020.

[55] R. Pelánek and J. Řihák, "Analysis and design of mastery learning criteria," *New Rev. of Hypermed. and Multimed.*, vol. 24, no. 3, pp. 133–159, 2018.

[56] P. Chandler and J. Sweller, "Cognitive load theory and the format of instruction," *Cogn. and Instruct.*, vol. 8, no. 4, pp. 293–332, 1991.

[57] R. Pelánek, T. Effenberger, and J. Čechák, "Complexity and difficulty of items in learning systems," *Int. J. of Artif. Intell. in Educ.*, 2021.

[58] R. G. Benjamin, "Reconstructing readability: Recent developments and recommendations in the analysis of text difficulty," *Educ. Psycholog. Rev.*, vol. 24, no. 1, pp. 63–88, 2012.

[59] V. Kuperman, H. Stadthagen-Gonzalez, and M. Brysbaert, "Age-of-acquisition ratings for 30,000 english words," *Behav. Res. Methods*, vol. 44, no. 4, pp. 978–990, 2012.

[60] R. Pelánek, "Measuring similarity of educational items: An overview," *IEEE Trans. on Learn. Technol.*, vol. 13, no. 2, pp. 354–366, 2020.

[61] Z. Kastrati, A. S. Imran, and A. Kurti, "Integrating word embeddings and document topics with deep learning in a video classification framework," *Pattern Recognit. Lett.*, vol. 128, pp. 85–92, 2019.

[62] R. Pelánek, "The details matter: methodological nuances in the evaluation of student models," *User Model. and User-Adapted Interact.*, vol. 28, no. 3, pp. 207–235, 2018.

[63] V. Chandola, A. Banerjee, and V. Kumar, "Outlier detection: A survey," *ACM Computing Surv.*, vol. 14, p. 15, 2007.

[64] R. Pelánek, "Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques," *User Model. and User-Adapted Interact.*, vol. 27, no. 3, pp. 313–350, 2017.

[65] R. S. d Baker, S. M. Gowda, M. Wixon, J. Kalka, A. Z. Wagner, A. Salvi, V. Aleven, G. W. Kusbit, J. Ocumpaugh, and L. Rossi, "Towards sensor-free affect detection in cognitive tutor algebra." in *Proc. Educational Data Mining*, 2012.

[66] C. G. Northcutt, A. D. Ho, and I. L. Chuang, "Detecting and preventing "multiple-account" cheating in massive open online courses," *Comput. & Educ.*, vol. 100, pp. 71–80, 2016.

[67] R. De Ayala, *The theory and practice of item response theory*. The Guilford Press, 2008.

[68] M. Streeter, "Mixture modeling of individual learning curves." in *Proc. Educational Data Mining*, 2015.

[69] T. Barnes, "The q-matrix method: Mining student response data for knowledge," in *Proc. Educational Data Mining Workshop*. AAAI Press, 2005, pp. 1–8.

[70] Y. Long, K. Holstein, and V. Aleven, "What exactly do students learn when they practice equation solving? refining knowledge components with the additive factors model," in *Proc. Learning Analytics and Knowledge*, 2018, pp. 399–408.

[71] B. Martin, A. Mitrovic, K. R. Koedinger, and S. Mathan, "Evaluating and improving adaptive educational systems with learning curves," *User Model. and User-Adapted Interact.*, vol. 21, no. 3, pp. 249–283, 2011.

[72] M. Desmarais, B. Beheshti, and P. Xu, "The refinement of a q-matrix: Assessing methods to validate tasks to skills mapping," in *Proc. Educational Data Mining*, 2014.

[73] Y. Chen, J. P. González-Brenes, and J. Tian, "Joint discovery of skill prerequisite graphs and student models." in *Proc. Educational Data Mining*, 2016.

[74] S. A. Adjei, A. F. Botelho, and N. T. Heffernan, "Predicting student performance on post-requisite skills using prerequisite skill data: an alternative method for refining prerequisite skill structures," in *Proc. Learning Analytics & Knowledge*, 2016, pp. 469–473.

[75] M. J. Brinkhuis, A. O. Savi, A. D. Hofman, F. Coomans, H. L. van Der Maas, and G. Maris, "Learning as it happens: A decade of analyzing and shaping a large-scale online learning system." *J. of Learn. Analytics*, vol. 5, no. 2, pp. 29–46, 2018.

[76] J. Čechák and R. Pelánek, "Item ordering biases in educational data," in *Proc. Artificial Intelligence in Education*, 2019, pp. 48–58.

[77] D. Hicks, M. Eagle, E. Rowe, J. Asbell-Clarke, T. Edwards, and T. Barnes, "Using game analytics to evaluate puzzle design and level progression in a serious game," in *Proc. Learning Analytics & Knowledge*, 2016, pp. 440–448.

[78] J. Papoušek, V. Stanislav, and R. Pelánek, "Evaluation of an adaptive practice system for learning geography facts," in *Proc. Learning Analytics & Knowledge*, 2016, pp. 134–142.

[79] P. Gyimesi, B. Vancsics, A. Stocco, D. Mazinanian, A. Beszédes, R. Ferenc, and A. Mesbah, "Bugsjs: a benchmark of javascript bugs," in *Proc. Software Testing, Validation and Verification*, 2019, pp. 90–101.

[80] R. Pelánek, "Beem: Benchmarks for explicit model checkers," in *International SPIN Workshop on Model Checking of Software*, 2007, pp. 263–267.

Radek Pelánek received a Ph.D. degree in computer science from Masaryk University, Brno, Czech Republic, for his work on formal verification. Since 2010 his research interests have focused on areas of educational data mining and learning analytics. Currently, he is the leader of the Adaptive Learning group at Masaryk University and is interested in both theoretical research in user modeling and the practical development of adaptive learning systems.

Tomáš Effenberger is a Ph.D. candidate in computer science at Masaryk University, Brno, Czech Republic, where he also received his master's degree. After research internships at the University of Oxford, Oxford, U.K. and Google AI, Zürich, Switzerland, he became a member of the Adaptive Learning group at Masaryk University. His mission is to make learning more efficient and engaging by advancing techniques for improving adaptive learning systems.