An Investigation of Challenges Encountered When Specifying Training Data and Runtime Monitors for Safety Critical ML Applications

Citation for the original published paper (version of record):

Heyn, H., Knauss, E., Malleswaran, I. et al (2023). An Investigation of Challenges Encountered
When Specifying Training Data and Runtime Monitors
for Safety Critical ML Applications. Lecture Notes in Computer Science (including subseries Lecture
Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 13975 LNCS: 206-222.
http://dx.doi.org/10.1007/978-3-031-29786-1_14

An Investigation of Challenges Encountered When Specifying Training Data and Runtime Monitors for Safety Critical ML Applications

Hans-Martin Heyn[1,2](✉), Eric Knauss[1,2], Iswarya Malleswaran[1],
and Shruthi Dinakaran[1]

[1] Chalmers University of Technology, SE-412 96 Gothenburg, Sweden
hans-martin.heyn@gu.se
[2] University of Gothenburg, SE-405 30 Gothenburg, Sweden

Abstract. **[Context and motivation]** The development and operation of critical software that contains machine learning (ML) models requires diligence and established processes. Especially the training data used during the development of ML models have major influences on the later behaviour of the system. Runtime monitors are used to provide guarantees for that behaviour. **[Question/problem]** We see major uncertainty in how to specify training data and runtime monitoring for critical ML models and by this specifying the final functionality of the system. In this interview-based study we investigate the underlying challenges for these difficulties. **[Principal ideas/results]** Based on ten interviews with practitioners who develop ML models for critical applications in the automotive and telecommunication sector, we identified 17 underlying challenges in 6 challenge groups that relate to the challenge of specifying training data and runtime monitoring. **[Contribution]** The article provides a list of the identified underlying challenges related to the difficulties practitioners experience when specifying training data and runtime monitoring for ML models. Furthermore, interconnection between the challenges were found and based on these connections recommendation proposed to overcome the root causes for the challenges.

Keywords: Artificial intelligence · Context · Data requirements · Machine learning · Requirements engineering · Runtime monitoring

1 Introduction

With constant regularity, unexpected and undesirable behaviour of machine learning (ML) models are reported in academia [9,24,26,53,54], the press, and

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 957197.

A. Ferrari and B. Penzenstadler (Eds.): REFSQ 2023, LNCS 13975, pp. 206–222, 2023.
https://doi.org/10.1007/978-3-031-29786-1_14

by NGOs[1]. These problems become especially apparent, and reported upon, when ML models violate ethical principles. Racial, religious, or gender biases are introduced through a lack of insight into the (sometimes immensely large set of) training data and missing runtime checks for example in large language models such as GPT-3 [1], or facial recognition software based on deep learning [37]. Unfortunately, improving the performance of deep learning models often requires an exponential growth in training data [3]. Data requirements can help in preventing unnecessarily large and biased datasets [50]. Due to changes in the environment, ML models can become "stale", i.e., the context changes so significantly that the performance of the model decreases below acceptable levels [5]. Runtime monitors collect performance data and indicate the need for re-training of the model with updated training data. However, these monitors need to be specified at design time. Data requirements can support the specification of runtime monitors [7]. The lack of specifications becomes specifically apparent with ML models that are part of *critical* software[2] because it is not possible to establish traceability from system requirements (e.g., functional safety requirements) to requirements set on the training data and the runtime monitoring [36].

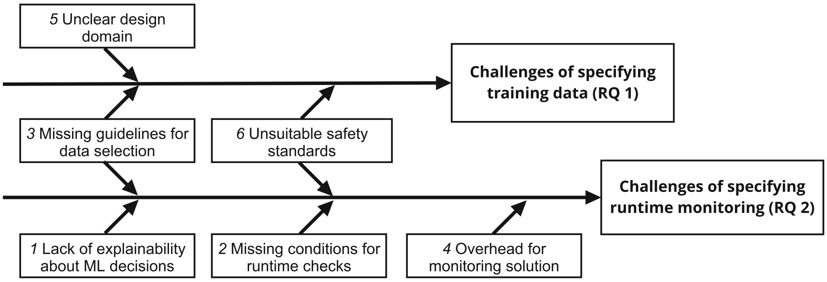
5 Unclear design domain
Challenges of specifying training data (RQ 1)
3 Missing guidelines for data selection
6 Unsuitable safety standards
Challenges of specifying runtime monitoring (RQ 2)
1 Lack of explainability about ML decisions
2 Missing conditions for runtime checks
4 Overhead for monitoring solution

Fig. 1. Overview of identified challenge categories

Scope and Research Questions

The purpose of this study is to highlight current challenges experienced by practitioners in specifying training data and runtime monitoring for ML in safety critical software.

The paper contributes a practitioner's point of view on the challenges reported in academic literature. The aim is to identify starting-points for a future engineering research on the use of runtime monitors for critical ML systems. The following research questions guided this study:

[1] Non-governmental organisations, e.g., https://algorithmwatch.org/en/stories/.

[2] We define critical software as software that is safety, privacy, ethically, and/or mission critical, i.e., a failure in the software can cause significant injury or the loss of life, invasion of personal privacy, violation of human rights, and/or significant economic or environmental consequences [31].

RQ1: What are challenges encountered by practitioners when specifying training data for ML models in safety critical software?

RQ2: What are challenges encountered by practitioners when specifying runtime monitors especially in relation to fulfilling safety requirements?

Figure 1 shows the main themes we found in answering the research questions. Concerning RQ1, the interviewees reported on several problems: the data selection process is nontransparent and guidelines especially towards defining suitable measures for data variety are missing. There are no clear context definitions that help in defining data needs, and current safety standards provide little guidance. Concerning RQ2, we found that the problem of defining suitable metrics and the lack of guidance from safety standards also inhibits the ability to specify runtime monitors. Furthermore, practitioners reported on challenges regarding explainability of ML decisions, and the processing and memory overhead caused by runtime monitors in safety critical embedded systems.

The remaining sections of this paper are structured as follows: Sect. 2 outlines and argues for the research methods of this study; Sect. 3 presents the results amd answers to the research questions; Sect. 4 discusses the findings, provides recommendations to practitioners and for further research, identifies related literature, elaborates on threats to validity, and provides a conclusion.

2 Research Method

We applied a qualitative interview-based survey with open-ended semi-structured interviews for data collection. Following the suggestions of Creswell and Creswell [13] the qualitative study was conducted in four steps: Preparation of interviews, data collection through interviews, data analysis, and result validation.

Preparations of Interviews. Based on the a-priori formulated research questions, two of the researchers of this study created an interview guide[3] which was validated and improved by the remaining two researchers. The interview guide contains four sections of questions: the first section includes questions about the interviewees' current role, background and previous experiences. The second section focuses on questions that try to understand challenges when specifying and selecting training data for ML models and how training data affect the performance of these models. The third section investigates challenges when ML models are incorporated in critical systems and how they affect the ability to specify training data. The fourth section concentrates on the run time monitoring aspect of the ML model and contains questions on challenges when specifying runtime monitors.

Sampling Strategy: We chose the participants for this study purposefully using a maximum variation strategy [14]. We were able to recruit interviewees from

[3] The interview guide is available at https://doi.org/10.7910/DVN/WJ8TKY.

five different companies, ranging from a local start-up to a multinational world leading communication company. An overview is given in Table 1.

A selection criteria for the company was that they must work with safety-critical systems and ML. Within the companies we tried to find interview candidates with different roles and work experiences to obtain a view beyond the developers' perspective. Besides function developers and ML model developers, we were interested in interviewing requirement engineers and product / function owners because they represent key roles in deriving system or function specifications. We provided the companies with a list of roles that we identified beforehand as interesting for interviewing[4]. Additionally, we interviewed two researchers from academia who participate in a joint industry EU Horizon 2020 project called VEDLIoT[5]. Both researchers worked also with ML models in industry before. Therefore, they could provide insights into both the academic and the industry perspective. A list of the ten interviewees for this study is provided in Table 2.

Table 1. Companies participating in the study

Company	Area of operations	Employees	Countries
1	Telecommunication networks	> 10.000	World
2	Automotive OEM	> 10.000	World
3	Automatic Driving	> 1.000	Europe
4	Industrial camera systems	> 1000	USA
5	Deep Learning optimisation for IoT	> 100	Sweden

Table 2. Participants of the study

Interviewee	Role	Experience
A	Researcher (Academic)	Functional Safety for ADAS (5 years)
B	Function developer	Sensor and perception systems (20 years)
C	Principal engineer	ML model integration (10 years)
D	ML model developer	Distributed and edge systems (3 years)
E	Function owner	ADAS perception functions (8 years)
F	Function developer and test engineer	Automatic driving systems (25 years)
G	Data Scientist	Distributed systems (12 years)
H	Requirement Engineer	Perception systems (8 years)
I	Researcher (Academic)	Neural Network development (8 years)
J	Functional Safety Manager	Sensor systems (20 years)

ADAS: Advanced Driver Assistance Systems

[4] The list included functional safety experts, requirement engineers, product owners or function owners, function or model developers, and data engineers.

[5] Very efficient deep learning in the Internet of Things.

Data Collection Through Interviews. All interviews were conducted remotely using either the conference software Zoom or Microsoft Teams and took between 60 - 90 min. The a-priori defined interview guide was only available to the interviewers and was not distributed to the participants beforehand. Each participant was interviewed by two interviewers who alternated in asking questions and observing. At the start of each interview, the interviewers provided some background information about the study's purpose. Then, the interview guide was followed. However, as we encouraged discussions with the interviewees, we allowed deviations from the interview guide by asking additional questions, or changing the order of the questions when it was appropriate [30]. All interviews were recorded and semi-automatically transcribed. The interviewers manually checked and anonymised the results.

Data Analysis. The data analysis followed suggestions by Saldana [42] and consisted of two cycles of coding and validation of the themes through a workshop and member checking.

First Coding Cycle: Attribute coding was used to extract information about the participants' role and previous experiences. Afterwards, the two interviewers independently applied structural coding to collect phrases in the interviews that represent topics relevant to answering the research questions. The researchers compared the individually assigned codes and applied descriptive coding with the aim of identifying phrases that describe common themes across the interviews.

Theme Validation: In a focus group, the identified themes were presented and discussed. Thirteen researchers from both industry and academia in the VEDLIoT project participated. Three of the participants also were interviewed for this study. The aim of the focus group was to reduce bias in the selection of themes and to identify any additional themes that the researchers might have missed.

Second Coding Cycle: After the themes were identified and validated, the second coding cycle was used to map the statements of the interviewees to the themes, and consequently identify the answers to the research questions. The second cycle was conducted by the two researchers who did not conduct the first cycle coding in order to reduce confirmation bias. The mapping was then confirmed and agreed upon by all involved researchers.

Result Validation. Member checking, as described in [14, Ch. 9] was used to validate the identified themes that answer RQ 1 and RQ 2. Additionally, we presented the results in a 60 min focus group to an industry partner and allowed for feedback and comments on the conclusions we drew from the data.

3 Results

During the first coding cycle, structural coding resulted in 117 statements for RQ1 and 77 statements for RQ2. Through descriptive coding preliminary themes were found. The statements and preliminary themes were discussed during a workshop. Based on the feedback from the workshop, 117 statements for RQ1 were categorised into eight final challenge themes and three challenge categories relating to the challenge of specifying training data. Similar, the 77 original statements for RQ2 were categorised into 13 final challenge themes in five challenge categories relating to the challenge of specifying runtime monitoring. A total of six challenge categories emerged for both RQs, out of which two categories contain challenges relating to both training data and runtime monitoring specification, and three challenge themes base on statements from both RQs. The categories and final challenge themes are listed in Table 3. Additionally, for each challenge theme, we indicate the implication of the findings for requirements engineering.

3.1 Answer to RQ1: Challenges Practitioners Experience When Specifying Training Data

The interviewees were asked to share their experiences in selecting training data, the influence of the selection of training data on the system's performance and safety, and any experiences and thoughts on defining specifications for training

Table 3. Challenge groups (bold) and themes found in the interview data. Data.: Challenges related to specifying training data (RQ1). Monitor.: Challenges related to specifying runtime monitoring (RQ2).

ID	Challenge Theme	Relates to		Related
		Data.	Monitor.	Literature
1	**Lack of explainability about ML decisions**		✓	
1.1	No access to inner states of ML models		✓	[18]
1.2	No failure models for ML models		✓	[51]
1.3	IP protection		✓	
2	**Missing conditions for runtime checks**		✓	
2.1	Unclear metrics and/or boundary conditions		✓	[11,21,43]
2.2	Unclear measure of confidence		✓	[17,34]
3	**Missing guidelines for data selection**	✓	✓	
3.1	Disconnection from requirements	✓		[16,42]
3.2	Grown data selection habits	✓		[20,33]
3.3	Unclear completeness criteria	✓		[49]
3.4	Unclear measure of variety	✓	✓	[45,50]
4	**Overhead for monitoring solution**		✓	
4.1	Limited resources in embedded systems		✓	[38]
4.2	Meeting timing requirements		✓	
4.3	Reduction of true positive rate		✓	
5	**Unclear design domain**	✓		
5.1	Design domain depends on available data	✓		[6]
5.2	Uncertainty in context	✓		[22]
6	**Unsuitable safety standards**	✓	✓	
6.1	Focus on processes instead of technical solution	✓	✓	[10]
6.2	No guidelines for probabilistic effects in software	✓		[28,43]
6.3	Safety case only through monitoring solution		✓	[31,46]

data for ML. Based on the interview data, we identified three challenge groups related to specifying training data: missing guidelines for data selection, unclear design domain, and unsuitable safety standards

Missing Guidelines for Data Selection. Four interviewees reported on a lack of guidelines and processes related to the selection of training data. A reason can be that data selection bases on "grown habits" that are not properly documented. Unlike conventional software development, the training of ML is an iterative process of discovering the necessary training data based on experience and experimentation. Requirements set on the data are described as disconnected and unclear for the data selection process. For example, one interviewee stated that if a requirements is set that images shall contain a road, it remains unclear what specific properties this road should have. Six interviewees described missing requirements on the data variety and missing completeness criteria as a reason for the disconnection of requirements from data selection.

> "**For example, we said that we shall collect data under varying weather conditions. What does that mean?**" - Interview B

> "**How much of it (the data) should be in darkness? How much in rainy conditions, and how much should be in snowy situations?**" - Interview F

Another interviewee stated that it is not clear how to measure variety, which could be a reason why it is difficult to define requirements on data variety.

> "**What [is] include[d] in variety of data? Is there a good measure of variety?**" - Interview A

RE Implication 1: **RE research should uncover new ways to specify variety and completeness criteria for data collection.**

Unclear Design Domain. Three interviewees describe uncertainty in the design domain as a reason for why it is difficult to specify training data. If the design domain is unclear, it will be challenging to specify the necessary training data.

> "**We need to understand for what context the training data can be used.**" - Interview J

> "**ODD [(Operational Design Domain)]? Yes, of course it translates into data requirements.**" - Interview F

RE Implication 2: **RE research must provide better ways to specify the context, since data selection and completeness criteria depend on it.**

Unsuitable Safety Standards. Because we were specifically investigating ML in safety critical applications, we asked the participants if they find guidance in safety standards towards specifying training data. Five interviewees stated that current safety standards used in their companies do not provide suitable guidance for the development of ML models. While for example ISO 26262 provides guidance on how to handle probabilistic effects in hardware, no such guidance is provided for software related probabilistic faults.

> **"The ISO 26262 gives guidance on the hardware design; [...] how many faults per hour [are acceptable] and how you achieve that. For the software side, it doesn't give any failure rates or anything like that. It takes a completely process oriented approach [...]."** - Interview C

One interviewee mentioned that safety standards should emphasise more the data selection to prevent faults in the ML model due to insufficient training.

> **"To understand that you have the right data and that the data is representative, ISO 26262 is not covering that right now which is a challenge."** - Interview H

RE Implication 3: **RE methods and practices are needed to operationalise safety standards for the selection of training data.**

3.2 Answer to RQ2: Challenges Practitioners Experience When Specifying Runtime Monitors

We asked the interviewees on the role of runtime monitoring for the systems they develop, their experience with specifying runtime monitoring, and the relation of runtime monitoring to fulfilling safety requirements on the system. We identified five challenge groups related to runtime monitoring: lack of explainability about ML decisions, missing conditions for runtime checks, missing guidelines for data selection, overhead for monitoring solution, and unsuitable safety standards.

Lack of Explainability About ML. A reason why it is difficult to specify runtime monitors for ML models is the inability to produce failure models for ML. In normal software development, causal cascades describe how a fault in a software components propagates trough the systems and eventually leads to a failure. This requires the ability to break down the ML model into smaller components and analyse their potential failure behaviour. Four interviewees however reported that they can only see the ML model as a "black-box" with no access to the inner states of the ML model. As a consequence, there is no insight into the failure behaviour of the ML model.

> **"[Our insight is] limited because it's a black box. We can only see what goes in and then what comes out to the other side. And so if there is some error in the behavior, then we don't know if it's because [of a] classification error, planning error, execution error?"** - Interview F

The reason for opacity of ML models is not necessarily due to technology limitations, but also due to constraints from protection of intellectual property (IP).

> **"Why is it a black box? That's not our choice. That's because we have a supplier and they don't want to tell us [all details]."** - Interview F

RE Implication 4: **RE can play a crucial role in navigating the trade-off between protecting IP of suppliers and sharing enough information to allow for safety argumentation.**

Missing Conditions for Runtime Checks. A problem of specifying runtime monitors is the need for finding suitable monitoring conditions. This requires the definition of metrics, goals and boundary conditions. Five interviewees report that they face challenges when defining these metrics for ML models.

> **"What is like a confidence score, accuracy score, something like that? Which score do you need to ensure [that you] classified [correctly]?"** - Interview F

Especially the relation between correct behaviour of the ML model and measure of confidence is unclear, and therefore impede runtime monitoring specification.

> **"We say confidence, that's really important. But what can actually go wrong here?"** - Interview J

RE Implication 5: **RE is called to provide methods for identifying conditions for runtime checks.**

Missing Guidelines for Data Selection. The inability to specify the meaning of data variety also relates to missing conditions for runtime checks. For example, runtime monitors can be used to collect additional training data, but without a measure of data variety it is difficult to find the required data points.

Overhead for Monitoring Solution. An often overlooked problem seems to be the induced (processing) overhead from a monitoring solution. Especially in the automotive sector, many software components run on embedded computer devices with limited resources.

> “**You don’t have that much compute power in the car, so the monitoring needs to be very light in its memory and compute footprint on the device, maybe even a separate device that sits next to the device.**” - Interview F

And due to the limited resources in embedded systems, monitoring solutions can compromise timing requirements of the system. Additionally, one interviewee reported concerns regarding the reduction of the ML model’s performance.

> “[...] **the true positive rate is actually decreasing when you have to pass it through this second opinion goal. It’s good from a coverage and safety point of view, but it reduces the overall system performance.**” - Interview F

RE Implication 6: **RE methods are needed to help finding suitable runtime checks that do not negatively impact the performance of the system.**

Unsuitable Safety Standards. Safety standards are mostly not suitable for being applied to ML model development. Therefore, safety is often ensured through non-ML monitoring solutions. Interviewees reported that it is not a good solution to rely only on the monitors for safety criticality:

> “[...] **so the safety is now moved from the model to the monitor instead, and it shouldn’t be. It should be the combination of the two that makes up safety.**” - Interview B

One reason is that freedom of inference between a non-safety critical component (the ML model), and a safety critical component (the monitor) must be ensured which can complicate the system design.

> “**And then especially if you have mixed critical systems [it] means you have ASIL [(Automotive Safety Integrity Level)] and QM [(Quality Management)] components in your design [and] you want to achieve freedom from interference in your system. You have to think about safe communication and memory protection.**” - Interview J

RE Implication 7: **RE is called to provide traceability and requirements information models that allow a complete description of the system, its monitors, and their relationship to high-level requirements (such as safety).**

4 Discussion and Conclusion

The results reveal connections between the challenges. Not all theme groups relate exclusively to one of the two challenges. For example, themes in the groups *unsuitable safety standards* and *missing guidelines for data selection* relate to both challenges of specifying training data and runtime monitoring. Furthermore, we identified cause-effect relations between different themes and across different group of themes. For example *IP protection* is a cause for the *inability of accessing the inner states* and for *creating failure models for ML model.* We based this assessment on a semantic analyses of the words used in the statements relating to these themes. For example, Interviewee F stated that:

> “**That neural network is something [of a] black box in itself. You don’t know why it do[es] things. Well, you cannot say anything about its inner behavior**” - Interview F

Later in the interview, the same participants states:

> “**Why is it a black box? That’s not our choice. That’s because we have a supplier and they don’t want to tell us [all details].**” - Interview F

Fig. 2 illustrates the identified cause-effect relations, relations between the themes, and how the different themes relate to the challenges.

Recommendations to Practitioners and for Further Research. The identified root causes of the challenges described by the participants allowed us to formulate recommendations listed in Table 4 and implications towards RE practises stated after each challenge theme in the previous section. Because these recommendations try to solve root causes described by the participants of the interview study, we think they are a useful first step towards solving the challenges related to specifying training data and runtime monitoring.

4.1 Related Literature

The problem of finding the “right” data: For acquiring data, data scientists have to rely on data mining with little to no quality checking and potential biases [4]. Biased datasets are a common cause for erroneous or unexpected behaviour of ML models in critical environments, such as in medical diagnostic [8], in the juridical system [19,38], or in safety-critical applications [15,47].

There are attempts to create “unbiased” datasets. One approach is to curate manually the dataset, such as in the FairFace dataset [29], the CASIA-SURF CeFaA dataset [33], or Fairbatch [41]. An alternative road is to use data augmentation techniques to “rebalance” the dataset [27,46]. However, it was discovered that it is not sufficient for avoiding bias to use an assumed balanced datasets during training [20,51,52] because it is often unclear which features in the data need to be balanced. Approaches for curating or manipulating the dataset require information on the target domain, i.e., one needs to set requirements on the dataset depending on the desired operational context [6,16,22]. But deriving a data specification for ML is not common practise [25,34,43].

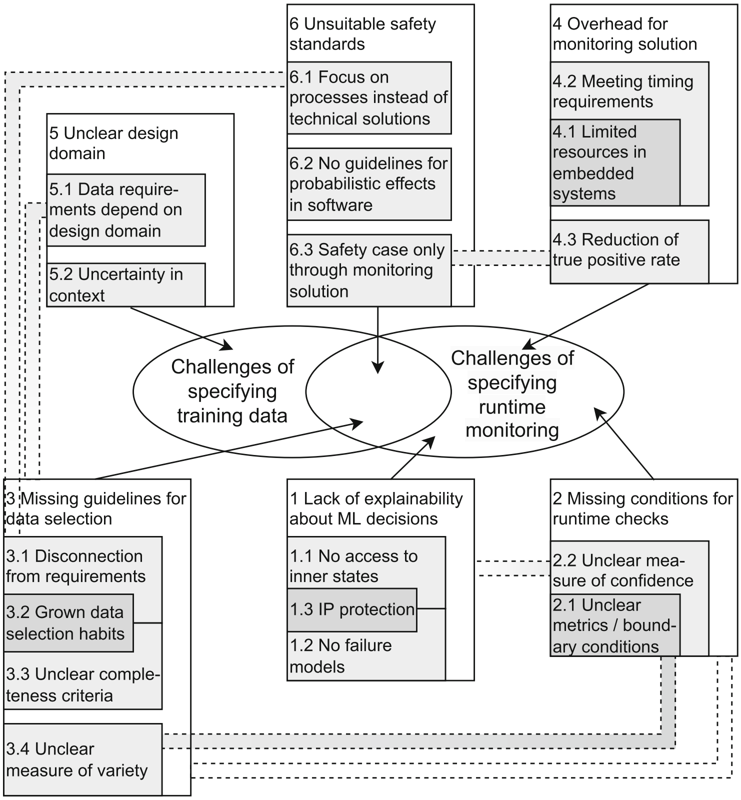
6 Unsuitable safety standards
6.1 Focus on processes instead of technical solutions
6.2 No guidelines for probabilistic effects in software
6.3 Safety case only through monitoring solution
4 Overhead for monitoring solution
4.2 Meeting timing requirements
4.1 Limited resources in embedded systems
4.3 Reduction of true positive rate
5 Unclear design domain
5.1 Data requirements depend on design domain
5.2 Uncertainty in context
Challenges of specifying training data
Challenges of specifying runtime monitoring
3 Missing guidelines for data selection
3.1 Disconnection from requirements
3.2 Grown data selection habits
3.3 Unclear completeness criteria
3.4 Unclear measure of variety
1 Lack of explainability about ML decisions
1.1 No access to inner states
1.3 IP protection
1.2 No failure models
2 Missing conditions for runtime checks
2.2 Unclear measure of confidence
2.1 Unclear metrics / boundary conditions

Fig. 2. Connection between the identified challenge themes. Enclosed themes have been identified as causes for the surrounding themes. Furthermore, dotted lines indicate relations between different themes.

The Problem of Finding the "Right" Runtime Monitor: Through clever test strategies, some uncertainty can be eliminated in regards to the behaviour of the model [11]. However, ML components are often part of systems of systems and their behaviour is hard to predict and analyse at design time [49]. DevOps principles from software engineering give promising ideas on how to tackle remaining uncertainty at runtime [35,48]. An overview of MLOps can be for example found in [32]. As part of the operation of the model, runtime models that "augment information available at design-time with information monitored at runtime" help in detecting deviations from the expected behaviour [17]. These runtime models for ML can take the form of model assertions, i.e., checking of explicitly defined attributes of the model at runtime [28]. However, the authors state that "bias in training sets are out of scope for model assertion". Another model based approach can be the creation of neuron activation patterns for runtime monitoring [12]. Other approaches treat the ML model as "black-box", and only check for anomalies and drifts in the input data [40] the output [44], or both

Table 4. Recommendations for practitioners and suggestions for further research

ID	Recommendation
I	**Avoid restrictive IP protection.** IP protection is a cause for the inability of accessing the inner states of the ML models (black-box model). This causes a nontransparent measure of confidence, and an inability to formulate failure models. To our knowledge, no studies have yet been performed on the consequences of IP protection of ML models on the ability to monitor and reason (e.g., in a safety case) for the correctness of ML model decisions.
II	**Relate measures of confidence to actual performance metrics.** For runtime monitoring, the measure of confidence is often used to evaluate the reliability of the ML model's results. But without understanding and relating that measure to clearly defined performance metrics of the ML model first, the measure of confidence provides little insight for runtime monitoring. In general, defining suitable metrics and boundary conditions should become an integral part of RE for machine learning as it affects both the ability to define data requirements and runtime monitoring requirements.
III	**Overcome grown data selection habits.** Grown data selection habits have been mentioned as a reason for a lack of clear completeness criteria and a disconnection from requirements. Based on our results, we argue that more systematic data selection processes need to be established in companies. This would allow for a better connection of the data selection process to requirement engineering and it creates a traceability between system requirements, completeness criteria and data requirements. Additionally, it might also reduce the amount of data needed for training, and therefore cost of development.
IV	**Balance hardware limitation in embedded systems.** Runtime monitoring causes a processing and memory overhead that can compromise timing requirements and reduce the ML model's performance. Today, safety criticality of systems with ML is mostly ensured through monitoring solutions. By decomposing the safety requirements instead onto both the monitoring and the ML model, the monitors might become more resource efficient, faster, and less constraining in regards to the decisions of the ML model. However, safety requirements on the ML models might trigger requirements on the training data.

[18]. However, similar to the aforementioned challenges when specifying data for ML, runtime monitoring needs an understanding on how to "define, refine, and measure quality of ML solutions" [23], i.e., in relation to non-functional requirements one needs to understand which quality aspects are relevant, and how to measure them [21]. Most commonly applied safety standards emphasise processes and traceability to mitigate systematic mistakes during the development of critical systems. Therefore, if the training data and runtime monitoring cannot be specified, a traceability between safety goals and the deployed system cannot be established [10].

For many researchers and practitioners, runtime verification and monitoring is a promising road to assuring safety and robustness for ML in critical software [2,11]. However, runtime monitoring also creates a processing and memory overhead that needs to be considered especially in resource-limited environments such as embedded devices [39].

The related work has been mapped to the challenges identified in the interview study in Table 3.

4.2 Threats to Validity

A lack of rigour (i.e., degree of control) in the study design can cause confounding which can manifest bias in the results [45]. The following mechanisms in this study tried to reduce confounding: The interview guide was peer-reviewed by an independent researcher, and a test session of the interview was conducted. To reduce personal bias, at least two authors were present during all interviews, and the authors took turn in leading the interviews. To confirm the initial findings from the interview study and reduce the risk of researchers' bias, a workshop was organised which was also visited by participants who were not part of the interview study. Another potential bias can arise from the sampling of participants. Although we applied purposeful sampling, we still had to rely on the contact persons of the companies to provide us with suitable interview candidates. We could not directly see a list of employees and choose the candidates ourselves. Regarding generalisability of the findings, the limited number of companies involved in the study can pose a threat to external validity. However, two of the companies are world-leading companies in their fields, which, in our opinion, gives them a deep understanding and experience of the discussed problems. Furthermore, we included companies from a variety of different fields to establish better generalisability. Furthermore, our data includes only results valid for the development of safety-critical ML models. We assume that the findings are applicable also to other forms of criticality, such as privacy-critical, but we cannot conclude on that generalisability based on the available data.

4.3 Conclusion

This paper reported on a interview-based study that identified challenges related to specifying training data needs and runtime monitoring for safety critical ML models. Through interviews conducted at five companies we identified 17 challenges in six groups. Furthermore, we performed a semantic analysis to identify the underlying root-causes. We saw that several underlying challenges affect both the ability to specify training data and runtime monitoring. For example, we concluded that restrictive IP protection can cause an inability to access and understand the inner states of a ML model. Without insight into the ML model's state, the measure of confidence cannot be related to actual performance metrics. Without clear performance metrics, it is difficult to define the necessary degree of variety in the training data. Furthermore, grown data selection impedes proper requirement engineering for training data. Finally, safety requirements should be distributed on both the ML model which can cause requirements on the training data, and on runtime monitors which can reduce the overhead by the monitoring solution. These recommendations will serve as starting point for further engineering research.

References

1. Abid, A., Farooqi, M., Zou, J.: Persistent anti-muslim bias in large language models. In: Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society, pp. 298–306 (2021)
2. Ashmore, R., Calinescu, R., Paterson, C.: Assuring the machine learning lifecycle: Desiderata, methods, and challenges. ACM Comput. Surv. **54**(5), 1–39 (2021)
3. Banko, M., Brill, E.: Scaling to very very large corpora for natural language disambiguation. In: Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, pp. 26–33 (2001)

4. Barocas, S., Selbst, A.D.: Big data's disparate impact. Calif. L. Rev. **104**, 671 (2016)
5. Bayram, F., Ahmed, B.S., Kassler, A.: From concept drift to model degradation: An overview on performance-aware drift detectors. Knowl. Based Syst. 108632 (2022)
6. Bencomo, N., Guo, J.L., Harrison, R., Heyn, H.M., Menzies, T.: The secret to better ai and better software (is requirements engineering). IEEE Softw. **39**(1), 105–110 (2021)
7. Bencomo, N., Whittle, J., Sawyer, P., Finkelstein, A., Letier, E.: Requirements reflection: requirements as runtime entities. In: Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, vol. 2, pp. 199–202 (2010)
8. Bernhardt, M., Jones, C., Glocker, B.: Potential sources of dataset bias complicate investigation of underdiagnosis by machine learning algorithms. Nat. Med. 1–2 (2022)
9. Blodgett, S.L., Barocas, S., Daum'e, H., Wallach, H.M.: Language (technology) is power: A critical survey of "bias" in nlp. In: ACL (2020)
10. Borg, M., et al.: Safely entering the deep: A review of verification and validation for machine learning and a challenge elicitation in the automotive industry. J. Automotive Softw. Eng. **1**(1), 1–19 (2018)
11. Breck, E., Cai, S., Nielsen, E., Salib, M., Sculley, D.: The ml test score: A rubric for ml production readiness and technical debt reduction. In: 2017 IEEE International Conference on Big Data, pp. 1123–1132. IEEE (2017)
12. Cheng, C.H., Nührenberg, G., Yasuoka, H.: Runtime monitoring neuron activation patterns. In: 2019 Design, Automation & Test in Europe Conference & Exhibition, pp. 300–303. IEEE (2019)
13. Creswell, J.W., Creswell, J.D.: Research design: Qualitative, quantitative, and mixed methods approaches. Sage publications (2017)
14. Creswell, John W.; Poth, C.N.: Qualitative Inquiry and Research Design: Choosing Among Five Approaches, 4th edn. Sage Publishing (2017)
15. Fabbrizzi, S., Papadopoulos, S., Ntoutsi, E., Kompatsiaris, I.: A survey on bias in visual datasets. arXiv preprint arXiv:2107.07919 (2021)
16. Fauri, D., Dos Santos, D.R., Costante, E., den Hartog, J., Etalle, S., Tonetta, S.: From system specification to anomaly detection (and back). In: Proceedings of the 2017 Workshop on Cyber-Physical Systems Security and PrivaCy, pp. 13–24 (2017)
17. Giese, H., et al.: Living with uncertainty in the age of runtime models. In: Bencomo, N., France, R., Cheng, B.H.C., Aßmann, U. (eds.) Models@run.time. LNCS, vol. 8378, pp. 47–100. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-08915-7_3
18. Ginart, T., Zhang, M.J., Zou, J.: Mldemon: Deployment monitoring for machine learning systems. In: International Conference on Artificial Intelligence and Statistics, pp. 3962–3997. PMLR (2022)
19. Goodman, B., Flaxman, S.: European union regulations on algorithmic decision-making and a "right to explanation". AI Mag. **38**(3), 50–57 (2017)
20. Gwilliam, M., Hegde, S., Tinubu, L., Hanson, A.: Rethinking common assumptions to mitigate racial bias in face recognition datasets. In: Proceedings of the IEEE CVF, pp. 4123–4132 (2021)
21. Habibullah, K.M., Horkoff, J.: Non-functional requirements for machine learning: understanding current use and challenges in industry. In: 2021 IEEE 29th RE Conference, pp. 13–23. IEEE (2021)

22. Heyn, H.-M., Subbiah, P., Linder, J., Knauss, E., Eriksson, O.: Setting AI in context: a case study on defining the context and operational design domain for automated driving. In: Gervasi, V., Vogelsang, A. (eds.) REFSQ 2022. LNCS, vol. 13216, pp. 199–215. Springer, Cham (2022). https://doi.org/10.1007/978-3-030-98464-9_16
23. Horkoff, J.: Non-functional requirements for machine learning: Challenges and new directions. In: 2019 IEEE 27th RE Conference, pp. 386–391. IEEE (2019)
24. Humbatova, N., Jahangirova, G., Bavota, G., Riccio, V., Stocco, A., Tonella, P.: Taxonomy of real faults in deep learning systems. In: 2020 IEEE/ACM 42nd International Conference on Software Engineering, pp. 1110–1121 (2020)
25. Ishikawa, F., Yoshioka, N.: How do engineers perceive difficulties in engineering of machine-learning systems?-questionnaire survey. In: 2019 IEEE/ACM Joint 7th International Workshop on Conducting Empirical Studies in Industry, pp. 2–9. IEEE (2019)
26. Islam, M.J., Nguyen, G., Pan, R., Rajan, H.: A comprehensive study on deep learning bug characteristics. In: 2019 ACM 27th European Software Engineering Conference, pp. 510–520 (2019)
27. Jaipuria, N., et al.: Deflating dataset bias using synthetic data augmentation. In: Proceedings of the IEEE CVF, pp. 772–773 (2020)
28. Kang, D., Raghavan, D., Bailis, P., Zaharia, M.: Model assertions for monitoring and improving ml models. Proc. Mach. Learn. Syst. **2**, 481–496 (2020)
29. Karkkainen, K., Joo, J.: Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation. In: Proceedings of the IEEE CVF, pp. 1548–1558 (2021)
30. King, N., Horrocks, C., Brooks, J.: Interviews in qualitative research. Sage (2018)
31. Knight, J.C.: Safety critical systems: challenges and directions. In: 24th International Conference on Software Engineering, pp. 547–550 (2002)
32. Kreuzberger, D., Kühl, N., Hirschl, S.: Machine learning operations (mlops): Overview, definition, and architecture. arXiv preprint arXiv:2205.02302 (2022)
33. Liu, A., Tan, Z., Wan, J., Escalera, S., Guo, G., Li, S.Z.: Casia-surf cefa: A benchmark for multi-modal cross-ethnicity face anti-spoofing. In: Proceedings of the IEEE CVF, pp. 1179–1187 (2021)
34. Liu, H., Eksmo, S., Risberg, J., Hebig, R.: Emerging and changing tasks in the development process for machine learning systems. In: Proceedings of the International Conference on Software and System Processes, pp. 125–134 (2020)
35. Lwakatare, L.E., Crnkovic, I., Bosch, J.: Devops for ai-challenges in development of ai-enabled applications. In: 2020 International Conference on Software, Telecommunications and Computer Networks, pp. 1–6. IEEE (2020)
36. Marques, J., Yelisetty, S.: An analysis of software requirements specification characteristics in regulated environments. J. Softw. Eng. Appli. (IJSEA) **10**(6), 1–15 (2019)
37. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., Galstyan, A.: A survey on bias and fairness in machine learning. ACM Comput. Surv. **54**(6), 1–35 (2021)
38. Miron, M., Tolan, S., Gómez, E., Castillo, C.: Evaluating causes of algorithmic bias in juvenile criminal recidivism. Artifi. Intell. Law **29**(2), 111–147 (2021)
39. Rabiser, R., Schmid, K., Eichelberger, H., Vierhauser, M., Guinea, S., Grünbacher, P.: A domain analysis of resource and requirements monitoring: Towards a comprehensive model of the software monitoring domain. Inf. Softw. Technol. **111**, 86–109 (2019)

40. Rahman, Q.M., Sunderhauf, N., Dayoub, F.: Per-frame map prediction for continuous performance monitoring of object detection during deployment. In: Proceedings of the IEEE CVF, pp. 152–160 (2021)
41. Roh, Y., Lee, K., Whang, S., Suh, C.: Sample selection for fair and robust training. Adv. Neural. Inf. Process. Syst. **34**, 815–827 (2021)
42. Saldaña, J.: The coding manual for qualitative researchers. Sage Publishing, 2nd edn. (2013)
43. Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., Aroyo, L.M.: "Everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In: 2021 Conference on Human Factors in Computing Systems, pp. 1–15 (2021)
44. Shao, Z., Yang, J., Ren, S.: Increasing trustworthiness of deep neural networks via accuracy monitoring. arXiv preprint arXiv:2007.01472 (2020)
45. Slack, M.K., Draugalis, J.R., Jr.: Establishing the internal and external validity of experimental studies. Am. J. Health Syst. Pharm. **58**(22), 2173–2181 (2001)
46. Uchôa, V., Aires, K., Veras, R., Paiva, A., Britto, L.: Data augmentation for face recognition with cnn transfer learning. In: 2020 International Conference on Systems, Signals and Image Processing, pp. 143–148. IEEE (2020)
47. Uricár, M., Hurych, D., Krizek, P., Yogamani, S.: Challenges in designing datasets and validation for autonomous driving. arXiv preprint arXiv:1901.09270 (2019)
48. Vierhauser, M., Rabiser, R., Grünbacher, P.: Requirements monitoring frameworks: A systematic review. Inf. Softw. Technol. **80**, 89–109 (2016)
49. Vierhauser, M., Rabiser, R., Grünbacher, P., Danner, C., Wallner, S., Zeisel, H.: A flexible framework for runtime monitoring of system-of-systems architectures. In: 2014 IEEE Conference on Software Architecture, pp. 57–66. IEEE (2014)
50. Vogelsang, A., Borg, M.: Requirements engineering for machine learning: Perspectives from data scientists. In: 2019 IEEE 27th International Requirements Engineering Conference Workshops, pp. 245–251. IEEE (2019)
51. Wang, A., et al.: Revise: A tool for measuring and mitigating bias in visual datasets. Int. J. Comput. Vis. 1–21 (2022)
52. Wang, T., Zhao, J., Yatskar, M., Chang, K.W., Ordonez, V.: Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (October 2019)
53. Wardat, M., Le, W., Rajan, H.: Deeplocalize: Fault localization for deep neural networks. In: 2021 IEEE/ACM 43rd International Conference on Software Engineering, pp. 251–262. IEEE (2021)
54. Zhang, X., et al.: Towards characterizing adversarial defects of deep learning software from the lens of uncertainty. 2020 IEEE/ACM 42nd International Conference on Software Engineering, pp. 739–751 (2020)