ISO/TC 22/SC 32/WG 14 "Safety and Artificial Intelligence"
Convenorship: **DIN**
Convenor: **Burton Simon Mr Prof. Dr.**

ISO PAS 8800 2024-01-31 for WG14 final check

Document type	Related content	Document date	Expected action
Project / Other	Project: ISO/CD PAS 8800	2024-01-31	**COMMENT/REPLY** by 2024-02-21

Description

Dear All,

attached is the final CD document (from the OSD).

Note for the ST leads: this is the same document that I distributed to you.

The document is without the updated drawings. The drawings are currently being reviewed.

As soon as we get back the updated drawings I will distriute the document again.

Please use this document for your final review and enter any comments you may have in the ISO comment sheet (N221).

If you have any comments, please send them to stephan.kraehnert@vda.de by 21 February.

Deadline: 2024-02-21

N222 contains the final WG14 comment sheet.

Please do not hesitate to contact me if you have any questions.

Best regards,

Stephan Kraehnert

ISO/TC 22/SC 32
ISO/CD PAS 8800(en)
Secretariat: JISC

Road Vehicles — Safety and artificial intelligence

Véhicules routiers — Sécurité et intelligence artificielle

© ISO 2024

All rights reserved. Unless otherwise specified, or required in the context of its implementation, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: + 41 22 749 01 11
E-mail: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Foreword ... viii
Introduction ... ix
1 Scope ... 1
2 Normative references ... 1
3 Terms and definitions ... 2
3.1 General AI-related definitions ... 2
3.2 Data-related definitions ... 9
3.3 General safety-related definitions ... 10
3.4 Safety: Root cause-, error-and failure-related definitions ... 13
3.5 Miscellaneous definitions ... 15
4 Abbreviated terms ... 18
5 Requirements for compliance ... 18
5.1 Purpose ... 18
5.2 General requirements ... 19
5.3 Interpretations of tables and figures ... 19
6 AI within the context of road vehicles system safety engineering and basic concepts ... 19
6.1 Application of the ISO 26262 series for the development of AI systems ... 19
6.2 Interactions with encompassing system-level safety activities ... 20
6.3 Mapping of abstraction layers between ISO 26262, ISO/IEC 22989 and this document ... 24
6.4 Example architecture for an AI system ... 27
6.5 Types of AI models ... 27
6.6 AI technologies of a ML model ... 28
6.7 Error concepts, fault models and causal models ... 29
6.7.1 Cause-and-effect chain ... 29
6.7.2 Root cause classes ... 30
6.7.3 Error classification based on the safety impact ... 31
7 AI safety management ... 32
7.1 Objectives ... 32
7.2 Prerequisites and supporting information ... 32
7.3 General requirements ... 33
7.4 Reference AI safety life cycle ... 35
7.5 Iterative development paradigms for AI systems ... 37
7.6 Work products ... 38
8 Assurance arguments for AI systems ... 39
8.1 Objectives ... 39
8.2 Prerequisites and supporting information ... 39
8.3 General requirements ... 40

© ISO 2024 – All rights reserved

8.4 AI system-specific considerations in assurance arguments ... 40
8.5 Structuring assurance arguments for AI systems ... 41
8.5.1 Context of the assurance argument ... 41
8.5.2 Categories of evidence ... 42
8.6 The role of quantitative targets and qualitative arguments ... 44
8.7 Evaluation of the assurance argument ... 45
8.8 Work products ... 46
9 Derivationof AI safetyrequirements ... 46
9.1 Objectives ... 46
9.2 Prerequisites and supporting information ... 46
9.3 General requirements ... 47
9.4 General workflow for deriving safety requirements ... 48
9.5 Deriving AI safety requirements on supervised machine learning ... 50
9.5.1 The need for refinedAI safety requirements ... 50
9.5.2 Derivation of refined AI safety requirements to manage uncertainty ... 52
9.5.3 Refinement of the input space definition for AI safety lifecycle ... 55
9.5.4 Restricting the occurrence of AI output insufficiencies ... 55
9.5.5 Metrics, measurements and threshold design ... 58
9.5.6 Considerations for deriving safety requirements ... 59
9.6 Work products ... 60
10 Selection of AI technologies, architectural and development measures ... 60
10.1 Objectives ... 60
10.2 Prerequisites ... 60
10.3 General requirements ... 60
10.4 Architecture and development process design or refinement ... 61
10.5 Examples of architectural and development measures for AI systems ... 62
10.6 Work products ... 66
11 Data-related considerations ... 66
11.1 Objectives ... 66
11.2 Prerequisites and supporting information ... 66
11.3 General requirements ... 66
11.4 Dataset life cycle ... 67
11.4.1 Datasets and the AI safety lifecycle ... 67
11.4.2 Reference dataset lifecycle ... 68
11.4.3 Dataset safety analysis ... 70
11.4.4 Dataset requirements development ... 76
11.4.5 Dataset design ... 78
11.4.6 Dataset implementation ... 79
11.4.7 Dataset verification ... 80

© ISO 2024 – All rights reserved

11.4.8 Dataset validation ... 81
11.4.9 Dataset maintenance ... 81
11.5 Work products ... 82
12 Verification and validation of AI system ... 82
12.1 Objectives ... 82
12.2 Prerequisites and supporting information ... 83
12.3 General requirements ... 83
12.4 AI/ML specific challenges to verification and validation ... 85
12.5 Verification and validation of the AI system ... 86
12.5.1 Scope of verification and validation of the AI system ... 86
12.5.2 AI component testing ... 88
12.5.3 Methods for testing the AI component ... 90
12.5.4 AI system integration and verification ... 92
12.5.5 Virtual testing vs physical testing ... 93
12.5.6 Evaluation of the safety-related performance of the AI system ... 94
12.5.7 AI systemsafety validation ... 95
12.6 Work products ... 95
13 Safety analysis of AI systems ... 95
13.1 Objectives ... 95
13.2 Prerequisites and supporting information ... 96
13.3 General requirements ... 97
13.4 Safety analysis of the AI system ... 97
13.4.1 Scope of the AI safety analysis ... 97
13.4.2 Safety analysis based on the results of testing ... 98
13.4.3 Safety analysis techniques ... 99
13.5 Work products ... 101
14 Measures during operation ... 101
14.1 Objectives ... 101
14.2 Prerequisites and supporting information ... 101
14.3 General requirements ... 102
14.4 Planning for operation and continuous assurance ... 102
14.4.1 Safety risk of the AI system during operation phase ... 102
14.4.2 Safety activities during the operation phase ... 103
14.5 Continual, periodic re-evaluation of the assurance argument ... 103
14.6 Measures to assure safety of the AI system during operation ... 104
14.6.1 General ... 104
14.6.2 Technical safety measures ... 104
14.6.3 Safe operation guidance and misuse prevention in the field ... 106
14.7 Field data collection ... 106

© ISO 2024 – All rights reserved

14.8 Evaluation and continuous development 108
14.8.1 Field risk evaluation 108
14.8.2 Countermeasures addressing field risk 109
14.8.3 AI re-training, re-validation, re-approval and re-deployment 109
14.9 Work products 110
15 Confidence in use of AI development frameworks and software tools used for AI model development 110
15.1 Objectives 110
15.2 Prerequisites and supporting information 110
15.3 General Requirements 110
15.4 Confidence in the use of AI development frameworks 111
15.5 Confidence in the use of tools used to support the AI-safety lifecycle 114
15.6 Principles for data-driven AI model training and evaluation 114
15.7 Work products 115
Annex A Overview and workflow of ISO PAS 8800 116
Annex B Example assurance argument structure for an AI-based vehicle function 121
B.1 General 121
B.2 Assurance argument pattern for supervised machine learning 121
B.3 Use of assurance claim points to increase confidence in the assurance argument 128
B.3.1 General remarks on the use of assurance claim points 128
B.3.2 Example assurance claim points to support assumptions or context: ACP-A2 for assumption A2. 129
B.3.3 Example assurance claim point to support inference: ACP-S1 for strategy S1 130
B.3.4 Example assurance claim point to support evidence: ACP-E5 130
Annex C ISO 26262:2018 Gap Analysis for ML 132
C.1 ISO 26262-4:2018 Tailoring and Guidance for ML 132
C.2 ISO 26262-6:2018 Tailoring for ML 132
Annex D Detailed considerations on safety-related properties of AI systems 139
Annex E STAMP/STPA example 142
E.1 Overview 142
E.2 STPA Example 142
E.2.1 STPA Step 1: Defining the purpose and scope of the analysis 142
E.2.2 STPA step 2: Modelling of the control structure 142
E.2.3 STPA step 3: Identification of unsafe control actions 143
E.2.4 STPA step 4: Identification of causal scenarios 143
E.2.5 Identifying safety measures to mitigate the safety-related issues 145
Annex F Identification of software units within NN-based systems 147
Annex G Architectural and Development Measures for AI Systems 150
G.1 Examples of architectural and development measures for AI systems 150
G.1.1 Measures for Architectural Redundancy 150

© ISO 2024 – All rights reserved

173 G.2 Qualitative and quantitative analysis of AI architectures 152
174 G.2.1 Identifying software units within AI architectures 153
175 G.3 Data distributions and their impacts on AI models 154
176 G.3.1 Out of distribution data and its mitigation 154
177 G.3.2 Distributional shift and its mitigation 154
178 G.4 Training safety measures 156
179 G.4.1 Hyperparameter tuning 156
180 G.4.2 Robust Learning 157
181 G.4.3 Transfer learning 157
182 G.4.4 Confidence calibration and uncertainty quantification of AI models 158
183 G.4.5 Verifying feature selection 158
184 G.4.6 Monitoring multiple scores 159
185 G.4.7 Attention or saliency Maps 159
186 G.4.8 Interpretable latent features 159
187 G.4.9 Augmentation of Data 159
188 G.5 Monitoring and AI system modification 161
189 G.5.1 Dynamic Environment Monitoring 161
190 G.5.2 AI Model Modification 161
191 G.6 Alignment of intention 162
192 G.7 Considerations related to the target execution environment 162
193 G.7.1 Optimization of parameters and optimization of architectural entities of AI components 163
194 G.7.2 Knowledge distillation also known as teacher-student model 163
195 G.7.3 Analysis for differences 163
196 Annex H Typical performance metrics for machine learning 164
197 Bibliography 171
198

© ISO 2024 – All rights reserved

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 22, Road Vehicles, Subcommittee SC 32, Electrical and electronic components and general system aspects, Working Group 14, Safety and Artificial Intelligence.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found atwww.iso.org/members.html.

© ISO 2024 – All rights reserved

Introduction

The purpose of this document is to provide industry-specific guidance on the use of AI systems in safety-related functions. It is not restricted to specific AImethods or specific vehicle functions.

This document defines a framework for managing AIsafety that tailors or extends existing approaches currently defined within the ISO 26262 series (functional safety) and ISO 21448 (safety of the intended functionality)[1].

Functional safety-related risks associated with malfunctioning behaviour of the AI system are addressed by tailoring or extending relevant clauses from ISO 26262-4:2018, ISO 26262-6:2018 and ISO 26262-8:2018.

Risks related to functional insufficiencies in the AI system are addressed by extending concepts and guidance provided by ISO 21448. A causal model of understanding the sources of functional insufficiencies in the AI system is proposed and used to derive a set of safety requirements on the AI system as well as a set of risk reduction measures.

NOTE 1 ISO 21448 is applicable to intended functionalities where proper situational awareness is essential to safety and where such situational awareness is derived from sensors and processing algorithms, especially functionalities of emergency intervention systems and systems having ISO/SAE PAS 22736[2]levels of driving automation from 1 to 5. Therefore, it is possible that systems utilize AI technologies that are formally not in scope of ISO 21448.

EXAMPLE 1 ISO 21448 will not apply to the development of an engine control unit that uses AI to optimize its performance while this document will.

This document recognizes that due to the wide range of applications of AI and associated safety requirements, as well as the rapidly evolving state-of-the-art, it is not possible to provide detailed requirements on the process or product characteristics required to achieve an acceptably low level of residual risk associated with the use of AI systems. Therefore, in addition to providing guidance for tailoring or extension of ISO 26262 and ISO 21448, this document focuses on the principles that are to be instantiated to support the creation of a project-specific assurance argument for the safety of the AI system within its vehicle context. This includes risk reduction measures during the design and operation phases using an iterative approach to reducing risk as outlined in IEC Guide 51 [3] is proposed.

The task of hazard and risk analysis is outside the scope of this document and is considered a part of vehicle level systems safety engineering activities as described in ISO 26262 and ISO 21448, or in application specific standards such as ISO TS 5083.

ISO/IEC TR 5469:2024 [4] provides generic guidance for the application of AI technologies as part of safety functions, independent of specific industry sectors. Many of the concepts outlined within ISO/IEC TR 5469 can be applied in the context of road vehicle. There is therefore a close relationship to concepts described within this document and ISO/IEC TR 5469.

ISO/IEC TR 5469 provides classification schemes to determine the safety requirements on the AI/ML function. These include the usage level and AI technology class.

The usage level is related to the nature of the task being performed by the engineered AI system.

NOTE 2 According to ISO/IEC TR 5469:

— Usage level A1 is assigned when the AI technology is used in a safety-relevant E/E/PE system and where automated decision-making of the system function using AI technology is possible;

— Usage level A2 is assigned when the AI technology is used in a safety-relevant E/E/PE system and where no automated decision-making of the system function using AI technology is possible (e.g. AI technology is used for diagnostic functionality within the E/E/PE system);

© ISO 2024 – All rights reserved

— Usage level B1 is assigned when the AI technology is used only during the development of the safety-relevant E/E/PE system (e.g. an offline support tool) and where automated decision-making of the function developed using AI technology is possible;

— Usage level B2 is assigned when the AI technology is used only during the development of the safety-relevant E/E/PE system (e.g. an offline support tool) and where no automated decision-making of the function is possible;

— Usage level C is assigned when the AI technology is not part of a functional safety function in the E/E/PE system, but can have an indirect impact on the function;

— Usage Level D is assigned if the AI technology is not part of a safety function in the E/E/PE system and has no impact on the safety function due to sufficient segregation and behaviour control.

The technology class is related to the problem complexity and the transferability of existing standards to demonstrating an adequate level of safety based on properties of the target function and the AI technology used.

NOTE 3 According to ISO/IEC TR 5469:

— Class I is assigned if the AI technology can be developed and reviewed using existing functional safety International Standards.

— Class II is assigned if AI technology cannot be fully developed and reviewed using existing functional safety International Standards, but it is still possible to identify the desired safety properties and the means to achieve them by a set of methods and techniques.

— Class III is assigned if AI technology cannot be developed and reviewed using existing functional safety International Standards and it is not possible to identify a set of properties with related methods and techniques to achieve them.

This document does not explicitly call out the classes and usage levels of ISO/IEC TR 5469.

EXAMPLE 2 For some AI technology, the application of ISO 26262 is deemed to be sufficient. This corresponds to the Class I of ISO/IEC TR 5469.

The guidance outlined within this document is relevant for all usage of AI for which safety requirements can foreseeably be allocated either through:

a) the use of AI for the functionality itself;

b) the use of AI as a safety mechanism.

NOTE 4 These usages correspond to the usage levels A1, A2, C of ISO/IEC TR 5469. In all cases, the applicability of the guidance provided within this document can be determined by the allocation of safety requirements to the AI technology, whereas the usage levels of ISO/IEC TR 5469 can be used to support the requirements elicitation process.

This document is aligned with standards and documents developed within ISO/IEC JTC1/SC42 Artificial Intelligence. AI specific definitions are used from ISO/IEC 22989 (Artificial intelligence — Artificial intelligence concepts and terminology), unless in conflict with safety-specific definitions.

Other documents developed within ISO/IEC JTC1/SC42 can be used to provide additional guidance on specific aspects of AI that are relevant to safety-related properties. Examples of such documents include ISO/IEC TR 24027:2021 (Artificial Intelligence (AI) – Bias in AI systems and AI aided decision making) and ISO/IEC TR 24029-1:2021 (Artificial Intelligence (AI) – Assessment of the robustness of neural networks – Part 1: Overview).

This document harmonizes the concepts already described in ISO 21448:2022, Annex D.2[1]and ISO/TS 5083,Annex B whilst extending these with specific guidance regarding the definition of safety requirements of ML, ML safety analyses and the creation of associated safety evidence during the development and deployment lifecycle.

© ISO 2024 – All rights reserved

ISO/TS 5083, Annex B[1] is an application of this document to Automated Driving Systems (ADS).

The relationship with the above-mentioned documents is summarized in Table 1–1.

Table 1–1 — How this document relates to other publications on AI safety

Publication	Relationship with this document
ISO/IEC 22989	AI specific definitions are used from ISO/IEC 22989, unless in conflict with safety-specific definitions. Safety-related properties are a subset of generic AI properties described in ISO/IEC 22989.
ISO/IEC TR 5469	This document does not explicitly call out the classes and usage levels of ISO/IEC TR 5469. This document considers and adapts to road vehicles the general framework described in ISO/IEC TR 5469 on safety properties, virtual testing and physical testing, confidence in use of AI development frameworks and architectural redundancy patterns.
ISO 26262	This document is a tailoring or extension of ISO 26262 for AI elements of the system. Refer to Clause 5 for details.
ISO 21448	This document is a tailoring or extension of ISO 21448 for AI elements of the system. Refer to Clause 5 for details.
ISO TS 5083	ISO TS 5083, Annex B**2** is an application of this document to Automated Driving Systems (ADS).

This document adds the following contents with respect to the mentioned publications:

— tailoring or extensions of ISO 26262 and ISO 21448 required specifically for AI elements of the system (referred to as AI systems);

— a conceptual model for reasoning about errors and their causes specific to AI systems;

— a reference AI safety lifecycle;

— the safety assurance argument for AI systems;

— a method for deriving AI safety requirements for AI systems;

— considerations for the design of safe AI systems;

— considerations on data management for the AI systems;

— a verification and validation strategy for AI systems;

— a safety analysis approach for AI systems (focused on insufficiencies);

— activities during operation required to ensure the continuous AI safety.

[1] Under preparation. Stage of the time of publication: ISO/FDISTS 5083.

© ISO 2024 – All rights reserved

Road Vehicles — Safety and artificial intelligence

1 Scope

This document is intended to be applied to safety-related systems that include one or more electrical and/or electronic (E/E) systems which use AI technology and that are installed in series production road vehicles, excluding mopeds. It does not address unique E/E systems in special vehicles such as E/E systems designed for drivers with disabilities.

This document addresses the risk of undesired safety-related behaviour at the vehicle level due to output insufficiencies, systematic error and random hardware error of AI elements within the vehicle. This includes the interaction with AI elements that are not part of the vehicle itself but can have a direct or indirect impact on the vehicle safety.

EXAMPLE 1 Examples of AI elements within the vehicle include the trained AI model and AI system.

EXAMPLE 2 Direct impact on safety can be due to e.g. object detection by elements external to the vehicle.

EXAMPLE 3 Indirect impact on safety can be due to e.g. field monitoring by elements external to the vehicle.

NOTE The development of AI elements that are not part of the vehicle is not within the scope of this document. These elements can comply with domain-specific safety guidance. This document can be used as a reference where such domain-specific guidance does not exist.

This document describes safety-related properties of AI systems that may be used to construct a convincing safety assurance claim for the absence of unreasonable risk.

This document does not provide specific guidelines for software tools that use AI methods.

This document focuses primarily on the subclass of AI methods defined as ML (Machine Learning): it covers the principles of established and well understood classes of ML, but is not focused on specific AI methods e.g. Deep Neural Networks (DNN).

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 21448:2022, *Road vehicles — Safety of the intended functionality*

ISO 21448:2022, *Road vehicles — Safety of the intended functionality*

ISO 21448:2022, *Road vehicles — Safety of the intended functionality*

ISO 21448:2022, *Road vehicles — Safety of the intended functionality*

ISO 21448:2022, *Road vehicles — Safety of the intended functionality*

ISO 26262-10:2018, *Road vehicles — Functional safety — Part 10: Guidelines on ISO 26262*

ISO 26262-2:2018, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-2:2018, *Road vehicles — Functional safety — Part 2: Management of functional safety*

ISO 26262-2:2018, *Road vehicles — Functional safety — Part 2: Management of functional safety*

© ISO 2024 – All rights reserved

ISO 26262-6:2018, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-6:2018, *Road vehicles — Functional safety — Part 6: Product development at the software level*

ISO 26262-8:2018, *Road vehicles — Functional safety — Part 8: Supporting processes*

ISO 34502:2022, *Road vehicles — Test scenarios for automated driving systems — Scenario based safety evaluation framework*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in Clauses 3.1, 3.2, 3.3, 3.4 and 3.5 apply. Also, the terms and definitions of ISO 26262-1, ISO 21448, ISO/IEC 22989, ISO/IEC TR 5469 apply unless redefined in this document.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at http://www.iso.org/obp

3.1 General AI-related definitions

3.1.1
AI component
element of an *AI system* (3.1.17)

EXAMPLE 1 *AI pre-processing* (3.1.11) component.

EXAMPLE 2 *AI post-processing* (3.1.9) component.

EXAMPLE 3 *AI model* (3.1.7).

EXAMPLE 4 conventional software component inside an *AI system* (3.1.17).

Note 1 to entry: AI components that are not *AI models* (3.1.7) or that do not contain *AI models* (3.1.7)are not developed according to this document. The integration of those components with AI components that are*AI models* (3.1.7)or that contain *AI models* (3.1.7)is performed according to this document.

Note 2 to entry: See 6.3 for an elaborationof the relationship of the different abstraction layers of ISO 26262:2018, ISO/IEC 22989:2022 and this document with each other.

[SOURCE: ISO/IEC 22989:2022- modified to be consistent with ISO 26262 definitions: Replaced "functional element" with "element", reworded to not use “construct”, added examples and Notes to entry.]

3.1.2
AI controllability
ability of an external agent to control the *AI element* (3.1.3), its output or the behaviour of the item influenced by the AI output in order to prevent harm

EXAMPLE Before setting a PWM signal of an actor determined by an *AI model* (3.1.7) it is limited by a simple threshold or a substitute approximate physical model by the consumer

Note 1 to entry: An external agent is a person or an element not belonging to the *AI system* (3.1.17).

© ISO 2024 – All rights reserved

3.1.3
AI element
AI component (3.1.1) or *AI system* (3.1.17)

Note 1 to entry: An AI element can refer to a subset of *components* (3.5.3) within an *AI system* (3.1.17) that provide related functionality.

Note 2 to entry: See 6.3 for an elaboration of the relationship of the different abstraction layers of ISO 26262:2018, ISO/IEC 22989:2022 and this document with each other.

3.1.4
AI explainability
property of an *AI system* (3.1.17) to express important factors influencing the *AI system* (3.1.17) results in a way that humans can understand

EXAMPLE The *AI system* (3.1.17) can be explainable by natural language or by visualizing feature attribution methods like gradient based heat/saliency maps or SHAP.

3.1.5
AI generalization
ability of an*AI model* (3.1.7) to adapt and perform well on previously unseen data during inference

3.1.6
AI method
type of *AI model* (3.1.7)

EXAMPLE 1 Deep neural network.

EXAMPLE 2 k-nearest neighbour.

EXAMPLE 3 Support vector machine.

3.1.7
AI model
construct containing logical operations, arithmetical operations or a combination of both to generate an inference or prediction based on input data or information without being completely defined by human knowledge

Note 1 to entry: Inference is using a model to understand the relation between predictors and a target. Prediction is to use a model to generate prediction (values close to the real seen or unseen targets) based on the inputs.

3.1.8
AI model validation
evaluation of the performance of different *AI model* (3.1.7)candidates through testing

Note 1 to entry: There are three terms, "AI model validation", "validation" and "safety validation", that are distinguished in this standard. AI model validation originates from the validation data used by the AI community, validation originates from classic system development and safety validation originates from ISO 26262.

Note 2 to entry: The AI model validation is executed using the AI validation dataset.

3.1.9
AI post-processing
any processing that is applied to the output of an *AI model* (3.1.7) for the purpose of mapping the raw output/s to a more contextually relevant and consumable format

© ISO 2024 – All rights reserved

EXAMPLE 1 A non-maximum suppression and thresholding for a bounding-box generation that serves to remove bounding boxes of low relevance and duplicates.

EXAMPLE 2 The outputs of a Mixture Density Networks are combined with a physical model (hybrid model).

Note 1 to entry: The AI post-processing would also include any data conversion that is used to bring the output into a common format for better comparability.

Note 2 to entry: The AI post-processing can have a positive or a negative impact on the safety-related properties of the output of the *AI system* (3.1.17).

3.1.10
AI predictability
ability of the *AI system* (3.1.17) to produce trusted predictions

Note 1 to entry: Trusted predictions means that the predications are accurate and that this claim is supported by statistical evidence.

3.1.11
AI pre-processing
any processing that is applied to the input of an *AI model* (3.1.7)

3.1.12
AI reliability
ability of the *AI element* (3.1.3)to perform the*AI task* (3.1.18) without *AI error* (3.4.1) under stated conditions and for a specified period of time

3.1.13
AI resilience
ability of the *AI element* (3.1.3)to recover and continue performing the *AI task* (3.1.18) after the occurrence of an *AI error* (3.4.1).

3.1.14
AI robustness
ability to maintain an acceptable level of performance under the presence of semantically insignificant, but reasonably expected changes to the input

EXAMPLE In image data these insignificant input changes might stem from naturally-induced image corruptions or sensor noise.

3.1.15
AI safety
absence of unreasonable *risk* (3.3.11) due to *AI errors* (3.4.1)caused by faults and functional insufficiencies

Note 1 to entry: This definition only applies in the context of this document. The term "AI safety" is commonly understood to have a broader meaning which includes ethics, value alignment, long-term considerations, etc.

3.1.16
AI safety requirement
safety requirement (3.3.16) of an *AI element* (3.1.3)

3.1.17
AI system
item or element that utilise one or more *AI models* (3.1.7)

© ISO 2024 – All rights reserved

EXAMPLE An AI system consisting out of the *AI component* (3.1.1) “Deep Neural Network for bounding box generation (*AI model* (3.1.7))” and of the *AI component* (3.1.1) “non-maximum suppression algorithm (*AI post-processing* (3.1.9)*AI component* (3.1.1).)”.

Note 1 to entry: The AI system can use various *AI methods* (3.1.6) and can utilize different *AI technologies* (3.1.19).

Note 2 to entry: The boundaries of the AI system are determined during the definition of AI system architecture.

Note 3 to entry: The AI system can contain one or more *AI components* (3.1.1).

Note 4 to entry: The term “AI system” serves in this document as the top level of abstraction of the content to be developed in compliance with the corresponding standard. As such it is possible in a distributed development that what one party considers to be an *AI component* (3.1.1)., the other party considers to be an AI system, as for them it represents the top level of the content they develop.

Note 5 to entry: See 6.3 for an elaboration of the relationship of the different abstraction layers of ISO 26262:2018, ISO/IEC 22989:2022 and this document with each other.

3.1.18
AI task
action required by the *AI element* (3.1.3) to achieve a specific goal

Note 1 to entry: Examples of AI tasks include classification, regression, ranking, clustering and dimensionalityreduction.

Note 2 to entry: the AI task can be seen as a semantic description of *AI model* (3.1.7).

[SOURCE: ISO/IEC 22989:2022 with following modifications: "task" -> "AI task" added "by the AI element", replacing "<artificial intelligence>"; modified notes to entries accordingly]

3.1.19
AI technology
any technology used within the lifecycle of an *AI system* (3.1.17) to design, develop, train, test, validate and implement the *AI model* (3.1.7)

EXAMPLE Examples of AI technologies are provided in6.6

3.1.20
AI testing
testing the *AI system* (3.1.17) or *AI model* (3.1.7)to estimate the expected performance and generalization capability in the field

Note 1 to entry: The AI testing is executed using an AI test dataset.

Note 2 to entry: Refer to ISO26262-1:2018, 3.169 "testing".

3.1.21
AI system safety validation
confirmation that an *AI safety requirement* (3.1.16) allocated to the *AI system* (3.1.17) is fulfilled

Note 1 to entry: In other standards validation denotes the check that requirements are suitable for intended use. Here the term is intentionally used in the different way that is common in the ML community, i.e. to verity the requirement implementation.

© ISO 2024 – All rights reserved

3.1.22
bias
undesired, systematic difference in the *AI systems* (3.1.17)' predictions with respect to particular classes of inputs in comparison to others due to potential incorrect learning process

EXAMPLE The classes of inputs can refer to images of objects and people in the context of computer vision.

Note 1 to entry: Bias can arise from an undesired systematic difference within the dataset, from limitations within the training process, or from limitations within the *AI model* (3.1.7) capability itself to accurately reflect the dataset.

[SOURCE: ISO/IEC 22989:2022, 3.5.4 modified - adapted the definition to the AI context, replaced Note 1 to entry, added example.]

3.1.23
control element
element (3.5.5) controlling the execution of the *AI task* (3.1.18)by the *AI element* (3.1.3)and other *AI element* (3.1.1)-related operations like updates

Note 1 to entry: The control element can control non-*AI elements* (3.1.3)as well.

3.1.24
data pre-processing
part of the AI workflow that transforms raw data such that it is usable as the input to create the *AI model* (3.1.7)

Note 1 to entry: Pre-processing can include reformatting, removal of outliers and duplicates, and ensuring the completeness of the data set.

3.1.25
encompassing system
item which contains the *AI System* (3.1.17)

3.1.26
ground truth
set of dataset annotations that are taken to be correct

Note 1 to entry: Individual annotations are derived from information external to the dataset.

Note 2 to entry: Individual annotations may be refined as new information becomes available.

[SOURCE: ISO/IEC 2382-37:2022(en), 37.09.34]

3.1.27
hyperparameter
parameters of the used *AI technologies* (3.1.19) that affect both the performance of the *AI model* (3.1.7) and its learning process

Note 1 to entry: Hyperparameters are selected prior to training and can be used in processes to help estimate *model parameters* (3.1.35).

Note 2 to entry: Examples of hyperparameters include the number of network layers, width of each layer, type of activation function, optimization method, learning rate for neural networks, the choice of kernel function in a support vector machine, number of leaves or depth of a tree, the number of clusters in K-means clustering, the maximum number of iterations of the expectation maximization algorithm and the number of Gaussians in a Gaussian mixture.

© ISO 2024 – All rights reserved

[SOURCE: ISO/IEC 22989:2022, 3.3.4 – modified: term has been redefined to be applicable to all kinds of AI methods, not only machine learning]

3.1.28
inference
reasoning by which conclusions are derived from known premises

Note 1 to entry: In AI, a premise is either a fact, a rule, a model, a feature, or raw data.

Note 2 to entry: The term "inference" refers both to the process and its result.

[SOURCE: ISO/IEC 22989:2022, 3.1.17]

3.1.29
input space
set of possible input values

Note 1 to entry: See *semantic input space* (3.1.37)and *syntactic input space* (3.1.39) for ways how an input space can be specified.

3.1.30
machine learning (ML)
process of optimizing *model parameters* (3.1.35)through computational techniques, such that the *model's* (3.1.34)behaviour aligns with data or experience and enables prediction beyond the training set

EXAMPLE Learning from experience can mean trying to represent non-static data like simulation, reinforcement learning environment, etc.

[SOURCE: ISO/IEC 22989:2022, 3.3.5 - modified: Replaced "reflects the data or experience" with "aligns with data or experience and enables prediction beyond the training set". Added EXAMPLE.]

3.1.31
ML algorithm
algorithm to optimize parameters of a *ML model* (3.1.32) from data according to given criteria

EXAMPLE Consider solving an univariate linear function $y = \theta_0 + \theta_1 x$ where y is an output or result, x is an input, θ_0 is an intercept (the value of y where x=0) and θ_1 is a weight. In machine learning, the process of determining the intercept and weights for a linear function is known as linear regression.

[SOURCE: ISO/IEC 22989:2022]

3.1.32
ML model
mathematical construct that generates an inference or prediction based on input data or information and comprises a functionality that is created by *machine learning* (3.1.30)

EXAMPLE If an univariate linear function ($y = \theta_0 + \theta_1 x$) has been trained using linear regression, the resulting model can be $y = 3 + 7x$.

Note 1 to entry: A ML model results from training based on a *ML algorithm* (3.1.31).

[SOURCE: ISO/IEC 22989:2022,3.3.7 modified - added "and comprises a functionality that is created by *machine learning*" to distinguish it from other mathematical constructs]

© ISO 2024 – All rights reserved

3.1.33
ML model training
iterative process to optimize a *ML model* (3.1.32)'s input and output behaviour on a given training data set with the intention to improve its quality (e.g. AI accuracy, *AI robustness* (3.1.14), generalization capability, run time), based on a *ML algorithm* (3.1.31) that can adapt *ML model* (3.1.32) parameters, *hyperparameter* (3.1.27)s, cost function or the model structures itself

[SOURCE: ISO/IEC 22989:2022,3.3.15- modified to elaborate the procedure and intention]

3.1.34
model
physical, mathematical, or otherwise logical representation of a system, entity, phenomenon, process or data

[SOURCE: ISO/IEC 22989:2022]

3.1.35
model parameter
internal variable of a *model* (3.1.34)that affects how it computes its outputs

Note 1 to entry: Examples of model parameters include the weights in a neural network and the transition probabilities in a Markov model.

[SOURCE: ISO/IEC 22989:2022]

3.1.36
safety-related AI element
AI element (3.1.3) that contributes to the achievement of an*AI safety requirement* (3.1.16) allocated to the AI system, that can contribute to the violation of an*AI safety requirement* (3.1.16) allocated to the AI system, or can contribute to both

Note 1 to entry:

3.1.37
semantic input space
set of possible input values on a semantic level

EXAMPLE The semantic input space acquired by a camera sensor can be described as consisting of street images containing lane markers of different colours, orientation and degradations that appear in different lighting and weather conditions.

Note 1 to entry: The semantic values correspond and conform to abstract semantic concepts expected within the input space.

3.1.38
semantic output space
set of possible output values on a semantic level

3.1.39
syntactic input space
set of possible input values on a syntactic level

EXAMPLE The syntactic input space acquired by a camera sensor can be described as an RGB image array of integers.

© ISO 2024 – All rights reserved

Note 1 to entry: The syntactic values can correspond and conform to the outputs produced by the low-level sensor output values.

3.1.40
syntactic output space
set of possible output values on a syntactic level

3.1.41
trained ML model
ML model (3.1.32) with a set of *model parameter* (3.1.35) as result of *model training* (3.1.33)

[SOURCE: ISO/IEC 22989:2022 modified - added "ML" as part of the term]

3.2 Data-related definitions

3.2.1
AI test dataset
dataset used to estimate the performance and generalization capability of an AI model or an AI system

Note 1 to entry: SeeClause 11 for more details

3.2.2
AI validation dataset
dataset used to compare the performance of different candidate *AI models* (3.1.7)

3.2.3
dataset insufficiency
insufficiency of the dataset regarding data-related safety properties under consideration

Note 1 to entry: Dataset insufficiency includes data integrity errors and data distribution errors

3.2.4
field monitoring dataset
dataset collected after the release of the *AI system* (3.1.17) while the product is in operation specifically used for field monitoring of the performance of the *AI system* (3.1.17)

3.2.5
hybrid dataset
dataset comprising data elements that are both real-world data elements and synthetic data elements

3.2.6
in distribution data
data whose features relevant for the *AI task* (3.1.18)are present and sufficiently well represented in the training dataset

Note 1 to entry: In distribution input does not guarantee correctness of AI model output.

3.2.7
metadata
data that provides additional information about the data element or dataset but is usually not directly involved in the training process

Note 1 to entry: Some metadata (e.g., ground truth) is also used for training.

© ISO 2024 – All rights reserved

3.2.8
out of distribution data
data containing features relevant for the *AI task* (3.1.18), either absent or not sufficiently well represented in the *training data set* (3.2.12), that can result in an *AI error* (3.4.1)

Note 1 to entry: Out of distribution (OOD) refers to data or inputs that fall outside the scope of what an AI or machine learning model was trained on or is designed to handle. When an AI system encounters OOD data, it may struggle to make accurate predictions or decisions because it lacks the necessary knowledge and experience to handle such inputs effectively. OOD data can lead to unexpected or unreliable model behaviour.

3.2.9
real-world dataset
dataset comprising data elements that have been created by real world acquisitions

3.2.10
safety-related KPI
key performance indicator relevant for the achievement of *AI safety* (3.1.15)

3.2.11
synthetic dataset
dataset comprising data elements that have been created artificially

Note 1 to entry: "created artificially" implies that the data was not directly collected from something that happened in the real world. Additionally, the data does not necessarily represent something that already happened in the real world.

3.2.12
training dataset
dataset used to train a*ML model* (3.1.32)

[SOURCE: ISO/IEC 22989:2022 definition that has been reworked to contain dataset]

3.3 General safety-related definitions

3.3.1
assurance
grounds for justified confidence that a claim has been or will be achieved

[SOURCE: ISO/IEC/IEEE 15026-1:2019]

3.3.2
assurance argument
reasoned, auditable artefact created supporting the contention that its top-level claim (or set of claims) is satisfied, including systematic arguments, its underlying evidences and explicit assumptions that support the claim(s)

Note 1 to entry: An assurance argument contains the following and their relationships:

— one or more claims about properties;

— arguments that logically link the evidence and any assumptions to the claim(s);

— a body of evidence and possibly assumptions supporting these arguments for the claim(s); and

— justification of the choice of top-level claim and the method of reasoning.

[SOURCE: ISO/IEC/IEEE 15026-1:2019 Modified: replaced "argumentation" with "arguments"]

© ISO 2024 – All rights reserved

3.3.3
claim
true-false statement about the limitations on the values of an unambiguously defined property — called the claim's property — and limitations on the uncertainty of the property's values falling within these limitations during the claim's duration of applicability under stated conditions

Note 1 to entry: Uncertainties may also be associated with the duration of applicability and the stated conditions.

Note 2 to entry: A claim potentially contains the following:

— property of the system-of-interest;

— limitations on the value of the property associated with the claim (e.g. on its range);

— limitations on the uncertainty of the property value meeting its limitations;

— limitations on duration of claim's applicability;

— duration-related uncertainty;

— limitations on conditions associated with the claim; and

— condition-related uncertainty.

Note 3 to entry: The term "limitations" is used to fit the many situations that can exist. Values can be a single value or multiple single values, a range of values or multiple ranges of values, and can be multi-dimensional. The boundaries of these limitations are sometimes not sharp, e.g. they can involve probability distributions and can be incremental.

[SOURCE: ISO/IEC/IEEE 15026-1:2019]

3.3.4
undesired safety-related behaviour at the vehicle level
hazardous behaviour, *RFIM prevention issue* (3.3.10) or malfunctioning behaviour that can cause a hazard

3.3.5
hazard
potential source of *harm*

[SOURCE: ISO 26262-1:2018, 3.75, modified — deleted "caused by malfunctioning behaviour (3.88) of the item (3.84)" and Note 1 to entry]

3.3.6
influencing factor
factor contributing to the achievement or the absence of a *safety-related property* (3.3.15)

3.3.7
misuse
usage in a way not intended by the manufacturer or the service provider

[SOURCE: ISO 21448:2022, 3.17 modified - Note to entries and examples where deleted. Some Notes to entries were changed into explicit definitions]

3.3.8
reasonably foreseeable
technically possible and with a credible or measurable rate of occurrence

© ISO 2024 – All rights reserved

Note 1 to entry: Expected misuse can be understood as a sub-class of reasonably foreseeable event.

[SOURCE: ISO 26262-1:2018]

3.3.9
reasonably foreseeable indirect misuse (RFIM)
reasonably foreseeable (3.3.8) misuse which leads to a reduced controllability of the hazardous behaviour, to a potentially increased severity of an occurring accident or a combination of both

[SOURCE: ISO 21448:2022, modified – term was introduced within Note 5 to entry of definition 3.17 misuse and now is explicitly defined]

3.3.10
RFIM prevention issue
inability to prevent or detect and mitigate a *RFIM* (3.3.9)

3.3.11
risk
combination of the probability of occurrence of *harm* and the severity of that *harm*

Note 1 to entry: Other forms of risk definitions exist, e.g., risk for other topics like the risk of a project to fail, etc. This document focuses on the risk regarding safety. Hence this definition was chosen.

Note 2 to entry: The resulting risk evaluation of an error of an AI component is typically equivalent to the evaluation of the potential to lead to a violation of a safety requirement allocated to the AI system. The evaluation can be quantitative as well as qualitative, depending on the safety requirement.

[SOURCE: ISO 26262-1:2018,3.128 – modified: Added Note 2 to entry]

3.3.12
safety
absence of unreasonable risk

[SOURCE: ISO 26262-1:2018]

3.3.13
AI safety measure
activity or technical solution to avoid, detect or control *AI errors* (3.4.1), to mitigate their harmful effects or a combination thereof

EXAMPLE AI safety analysis.

Note 1 to entry: Safety measures include architectural measures.

Note 2 to entry: The AI safety measures include ISO 26262 safety measures of *AI elements* (3.1.3)as well as measures to address functional insufficiencies in compliance with ISO 21448 (e.g. functional modifications addressing SOTIF-related risks).

3.3.14
safety validation
assurance, based on examination and tests, that the safety goals are adequate and have been achieved with a sufficient level of integrity

Note 1 to entry: There are three terms, "AI model validation", "validation" and "safety validation", that are distinguished in this document. AI model validation originates from the validation data used by the AI community, validation originates

© ISO 2024 – All rights reserved

from classic system development and safety validation originates from ISO 26262. The three validation meanings are not the same.

[SOURCE: ISO 26262-1:2018, modified - Deleted note 1 to entry, added new Note 1 to entry]

3.3.15
safety-related property
property impacting safety

3.3.16
safety requirement
requirement related to safety

EXAMPLE 1 SOTIF requirement.

EXAMPLE 2 Functional safety requirement.

EXAMPLE 3 Technical safety requirement.

Note 1 to entry: This includes, but is not limited to, safety requirements motivated by functional safety as well as SOTIF.

3.3.17
unreasonable risk
risk judged to be unacceptable in a certain context according to valid societal moral concepts

[SOURCE: ISO 26262-1:2018]

3.3.18
work product
work product of the safety lifecycle that can be used as evidence within a safety assurance argument

3.4 Safety: Root cause-, error-and failure-related definitions

3.4.1
AI error
one or more discrepancies between computed, observed or measured values or conditions of the AI element, and the true, specified or theoretically correct values or conditions of the AI element

Note 1 to entry: An AI error can be a single discrepancy or a sequence of discrepancies.

Note 2 to entry: An AI error can be an error caused by a fault. Faults are typically addressed by ISO 26262.

Note 3 to entry: An AI error can be an output insufficiency caused by a functional insufficiency

3.4.2
AI triggering condition
specific conditions of a scenario that serve as an initiator for a subsequent *AI error* (3.4.1)

Note 1 to entry: *Functional insufficiencies* (3.4.6) or *faults* (3.4.5) are themselves not AI triggering conditions but are potentially activated by them thus leading to the occurrence of an *AI error* (3.4.1).

3.4.3
contributing AI error
AI error (3.4.1) which can lead to a violation of an*AI safety requirement* (3.1.16)allocated to the *AI system* (3.1.17), either by itself or in combination with one or more other *AI errors* (3.4.1).

© ISO 2024 – All rights reserved

3.4.4
AI error rate
probability density of *AI error* (3.4.1)occurrence divided by probability of no *AI error* (3.4.1) occurring until the measuring point

Note 1 to entry: Measurement units can include errors per hour, errors per km, etc.

Note 2 to entry: This is an analogue definition to the failure rate.

3.4.5
fault
abnormal condition that can cause an element or an item to fail

Note 1 to entry: Permanent, intermittent, and transient faults (especially soft errors) are considered.

Note 2 to entry: When a subsystem is in an error state it could result in a fault for the system.

Note 3 to entry: An intermittent fault occurs from time to time and then disappears again. This type of fault canoccur when a component is on the verge of breaking down or, for example, due to an internal malfunction in a switch. Some *systematic fault* (3.4.13)s (e.g. timing irregularities) could lead to intermittent faults.

[SOURCE: ISO 26262-1:2018, 3.54]

3.4.6
functional insufficiency
insufficiency of specification (3.4.7) or performance insufficiency

Note 1 to entry: A functional insufficiency activated by a *triggering condition* leads per definition to either an output insufficiency, a hazardous behaviour, a RFIM prevention issue or a combination of these.

[SOURCE: ISO 21448:2022, 3.8, modified – Examples, figures and notes to entry have been removed. A new Note to entry has been added as a replacement of Note 2 to entry]

3.4.7
insufficiency of specification
specification, possibly incomplete, contributing to either a hazardous behaviour or an *RFIM prevention issue* (3.3.10) when activated by one or more triggering conditions

Note 1 to entry: An insufficiency of specification activated by a triggering condition leads per definition to either an output insufficiency, a hazardous behaviour, an *RFIM prevention issue* (3.3.10) or a combination of these.

Note 2 to entry: More details can be found in Clause 6.7.1.

[SOURCE: ISO 21448:2022, 3.12, modified – Examples, notes to entry have been removed. A new Note to entry has been added for clarification]

3.4.8
output insufficiency
incorrect output of an element as a result of a *triggering condition* activating a *functional insufficiency* (3.4.6) of the element, contributing to either a hazardous behaviour, a *RFIM prevention issue* (3.3.10) or both

[SOURCE: ISO 21448:2022, modified – term was introduced within Note to entry 6 of definition 3.8 functional insufficiency and now is explicitly defined]

© ISO 2024 – All rights reserved

3.4.9
random hardware fault
hardware fault with a probabilistic distribution

[SOURCE: ISO 26262-1:2018, 3.119]

3.4.10
safety-related AI error
AI error (3.4.1) of a safety-related *AI element* (3.1.3)

3.4.11
safety-related fault
fault of a safety-related *AI element* (3.1.3)

3.4.12
systematic error
error due to a systematic fault

3.4.13
systematic fault
fault (3.4.5) whose failure is manifested in a deterministic way that can only be prevented by applying process or design measures

[SOURCE: SOURCE: ISO 26262-1:2018, 3.165]

3.5 Miscellaneous definitions

3.5.1
architecture
representation of the structure of the item or element that allows identification of building blocks, their boundaries and interfaces, and includes the allocation of requirements to these building blocks

[SOURCE: ISO 26262-1:2018]

3.5.2
architectural measure
technical solution implemented by the *AI element* (3.1.3) to detect and mitigate or tolerate *AI errors* (3.4.1) in order to uphold the ability to execute the *AI task* (3.1.18) in a safe manner or to achieve or maintain a dedicated operating mode in case of *AI errors* (3.4.1) without unreasonable *risk* (3.3.11)

EXAMPLE 1 Addition of output layers in the AI model for classification. AI models can make incorrect predictions that can lead to hazardous behaviour. Therefore, it would be beneficial for a model to be cautious in situations where it is uncertain about its predictions. One way to accomplish this is to design AI models by adding output layer(s) to represent reject class or reject option. Such models assess their confidence in each prediction and have the option to abstain from making a prediction when they are likely to make incorrect prediction.

EXAMPLE 2 Addition of redundant *AI components* (3.1.1).

Note 1 to entry: Architectural measure has a tangible impact on the *AI system* (3.1.17)or *AI component* (3.1.1)and lead to enhancement or modification of the architecture of *AI system* (3.1.17) or *AI component* (3.1.1).

Note 2 to entry: an architectural artefact that is used during development, for example using saliency map to argue the explainability of the system, but removed for the system deployment, those measures are NOT considered architectural measures.

© ISO 2024 – All rights reserved

3.5.3
component
non-system level element that is logically or technically separable and is comprised of more than one *hardware part* (3.5.6) or one or more *software unit* (3.5.10) or a combination of hardware part(s) and software unit(s)

EXAMPLE A microcontroller.

Note 1 to entry: A component is a part of a *system* (3.5.11).

[SOURCE: ISO 26262-1:2018 - modified: added "or a combination of hardware part(s) and software unit(s)"]

3.5.4
development measure
Appropriate process step(s) (activity) for the development of an *AI system* (3.1.17) or an *AI component* (3.1.1) that facilitates fulfilling *AI safety requirement* (3.1.16)s and/or enhancing the AI properties.

Note 1 to entry: Analysis of AI System or AI component, specific activity used during or after the training of the AI component can be a development measure. Please refer to 10.4for more details.

3.5.5
element
system, components (system, hardware or software), hardware parts, or software units

Note 1 to entry: When “software element” or “hardware element” is used, this phrase denotes an element of software only or an element of hardware only, respectively.

[SOURCE: ISO 26262-1:2018 - modified: added "system" to the components and removed Note 2 to entry]

3.5.6
hardware part
portion of a hardware component at the first level of hierarchical decomposition

EXAMPLE The CPU of a microcontroller, a resistor, flash array of a microcontroller.

[SOURCE: ISO 26262-1:2018]

3.5.7
item
system or combination of systems, to which ISO 26262 is applied, that implements a function or part of a function at the vehicle level

[SOURCE: ISO 26262-1:2018 - modified: Removed Note 1 to entry]

3.5.8
off-board
property indicating that a given task is done external to the vehicle system

3.5.9
on-board
property indicating that a given task is done internal to the vehicle system

3.5.10
software unit
atomic level software component of the software architecture that can be subjected to stand-alone testing

© ISO 2024 – All rights reserved

[SOURCE: ISO 26262-1:2018]

3.5.11
system
set of components or subsystems that relates at least a sensor, a controller and an actuator with one another

Note 1 to entry: The related sensor or actuator can be included in the system or can be external to the system.

[SOURCE: ISO 26262-1:2018]

3.5.12
testing
process of planning, preparing, and operating or exercising an item or element to verify that it satisfies specified requirements, to detect safety anomalies, to validate that requirements are suitable in the given context and to create confidence in its behaviour

Note 1 to entry: "to create confidence in its behaviour" includes also the evaluation of the performance of the element or item.

[SOURCE: ISO 26262-1:2018, modified - added Note to entry]

3.5.13
validation
confirmation, through the provision of objective evidence, that the requirements for a specific intended use or application have been fulfilled

Note 1 to entry: There are three terms, "AI model validation", "validation" and "safety validation", that are distinguished in this standard. AI model validation originates from the validation data used by the AI community, validation originates from classic system development and safety validation originates from ISO 26262. The three validation meanings are not the same.

[SOURCE: ISO/IEC 22929:2022]

3.5.14
verification
confirmation, through the provision of objective evidence, that specified requirements have been fulfilled

EXAMPLE The typical verification activities can be classified as follows:

— verification review, walk-through, inspection;

— verification testing;

— simulation;

— prototyping; and

— analysis (safety analysis, control flow analysis, data flow analysis, etc.)

Note 1 to entry: Verification only provides assurance that a product conforms to its specification.

[SOURCE: ISO/IEC 22989:2022]

© ISO 2024 – All rights reserved

4 Abbreviated terms

ACP	assurance claim point
ADS	automated driving system
AI	artificial intelligence
ASIL	automotive safety integrity level
DLC	dataset lifecycle
DNN	deep neural network
E/E	electrical/electronic
FMEA	failure mode and effects analysis
FN	false negative
FP	false positive
FPS	frames per second
GSN	goal structuring notation
HARA	hazard analysis and risk assessment
HAZOP	hazard and operability study
HMI	human machine interface
KPI	key performance indicator
ID	in distribution
ML	machine learning
NN	neural network
ODD	operational design domain
OOD	out of distribution
OTA	over the air
PFD	probability of failure on demand
RFDM	reasonably foreseeable direct misuse
RFIM	reasonably foreseeable indirect misuse
SOTIF	safety of the intended functionality
TN	true negative
TP	true positive

5 Requirements for compliance

5.1 Purpose

This clause describes how:

a) to achieve compliance with this document;

b) to interpret the applicability of each clause; and

© ISO 2024 – All rights reserved

c) to interpret the tables and figures used in this document.

5.2 General requirements

When claiming compliance with this document, each normative requirement shall be met, unless one of the following applies:

a) tailoring of the safety activities as defined in Clause 6 or in accordance with ISO 26262-2 has been performed that shows that the requirement does not apply; or

b) a rationale is available that the non-compliance is acceptable and the rationale has been evaluated in accordance with this document and ISO 26262-2, when applicable.

Informative content, including notes and examples, is only for guidance in understanding, or for clarification of the associated requirement, and shall not be interpreted as a requirement itself or as complete or exhaustive.

The results of safety activities are given as work products. “Prerequisites” are information which shall be available as work products of a previous phase or from an external source. Given that certain requirements of a clause are ASIL dependent or may be tailored, certain work products may not be needed as prerequisites.

5.3 Interpretations of tables and figures

Tables and figures can be normative or informative depending on their context. Tables and figures that are referenced by normative requirements are considered normative unless it is explicitly specified otherwise. Other tables and figures are only informative.

In case it is possible to fulfil a requirement with a different combination of methods, a rational is provided that the chosen combination of methods fulfil the requirement.

6 AI within the context of road vehicles system safety engineering and basic concepts

6.1 Application of the ISO 26262 series for the development of AI systems

This document isintended to be applied in combination with the ISO 26262 seriesto specifically address the safety of AI systems.

— For AI components thatare not AI models, or do not contain AI models, the ISO 26262 series can be applied by itself.

— For AI components that are AI models, or that contain AI models, the ISO 26262 series can be tailored and applied in combination with this document (see Figure 6–1).

© ISO 2024 – All rights reserved

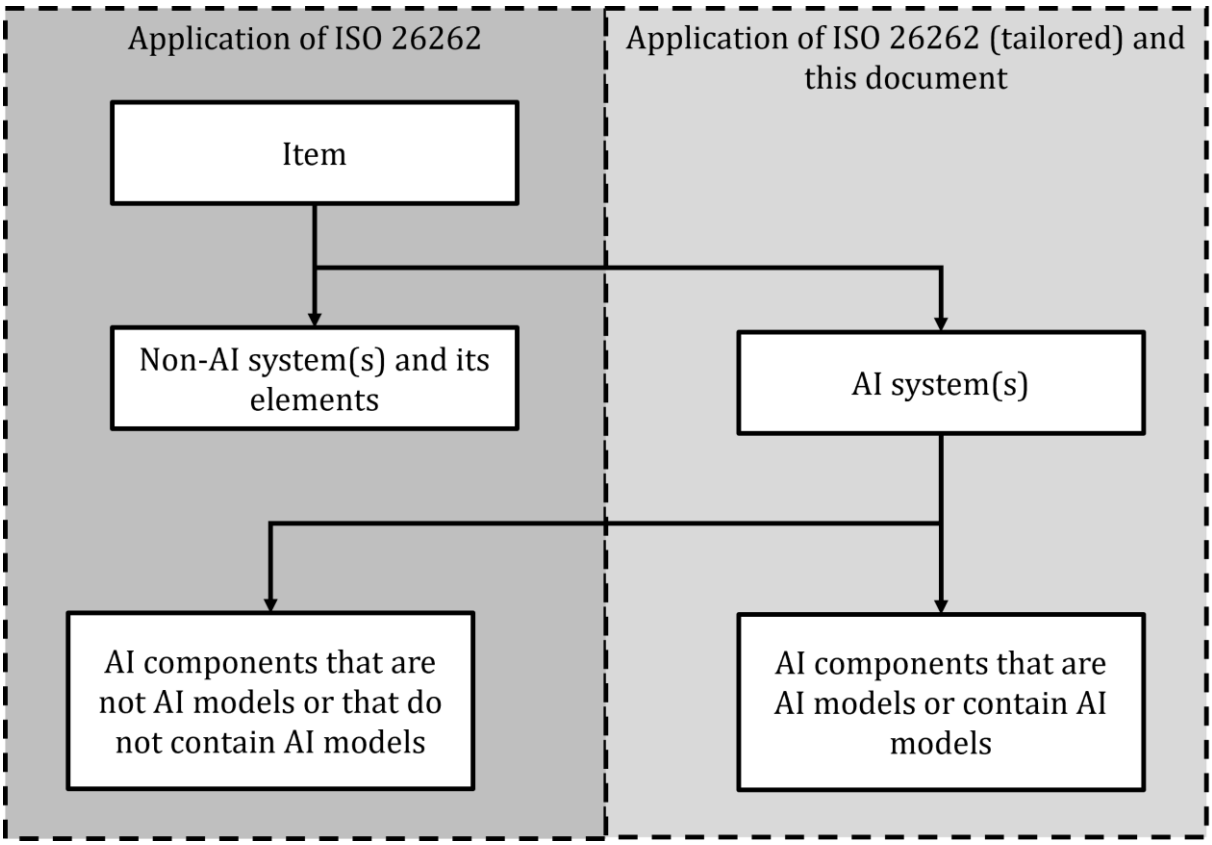


Figure 6–1 — Visualization of the applicability of the ISO 26262 series and this document to the item and its elements

NOTE See Annex C for a possible tailoring of ISO 26262-4:2018 and ISO 26262-6:2018 for ML.

6.2 Interactions with encompassing system-level safety activities

An example interaction of the AI system development with the encompassing system development based on the ISO 26262 series and ISO 21448 can be found in Table 6–1 and in Table 6–2. These tables also include remarks regarding the interaction and the applicability of the corresponding standards with this document and the development of the AI elements.

NOTE 1 ISO 21448 is applicable to intended functionalities where proper situational awareness is essential to safety and where such situational awareness is derived from complex sensors and processing algorithms, especially functionalities of emergency intervention systems and systems having SAE levels of driving automation from 1 to 5. However, this document is applicable to all AI systems which errors can impact safety independent of the vehicle-level functionality. It is possible that systems utilize AI technologies that are not in scope of ISO 21448, resulting in the applicability of this document but not ISO 21448.

NOTE 2 Although this document and ISO 21448 both focus on functional insufficiencies, the compliance with one does not automatically imply compliance with the other.

During the encompassing system architecture design phase, encompassing system requirements are decomposed and allocated to the AI systems as well as other elements of the encompassing system.

Table 6–1 — Example interaction of the AI element development in compliance with this document with the ISO 26262 series

© ISO 2024 – All rights reserved

ISO 26262:2018[a]	**Interaction of the AI system with the encompassing system (the AI system is not an item)**	**AI system: System component consisting of hardware and software components**	**AI component: Conventional[b] hardware component**	**AI component: AI model implemented by software component(s)**
2-5 Overall safety management	-	Adapted to address also the management of AI safety	Directly applicable	Adapted to address also the management of AI safety
2-6 Project dependent safety management	The management of the AI safety is part of the safety management of the encompassing system	Adapted to address also the management of AI safety	Directly applicable	Adapted to address also the management of AI safety
2-7 Safety management regarding production, operation, service and decommissioning	The management of the AI safety is part of the safety management of the encompassing system	Adapted to address also the management of AI safety	Directly applicable	Adapted to address also the management of AI safety
3-5 Item definition	Potential source of input space specification	-	-	-
3-6 Hazard analysis and risk assessment	-	-	-	-
3-7 Functional safety concept	Potential source of safety requirements allocated to the AI system	-	-	-
4-6 Technical safety concept	Potential source of safety requirements allocated to the AI system	Applicable (tailoring can be necessary)	Hardware safety requirements are derived from technical safety requirements allocated to the AI system	Software safety requirements are derived from technical safety requirements allocated to the AI system
4-7 System and item integration and testing	AI system as a system component to be integrated into the encompassing system	Integration of the hardware and software components of the AI system (tailoring can be necessary)	-	-
4-8 Safety validation	potential source of additional validation strategies and requirements	-	-	-
Part 5: Product development at the hardware level	Potential source of hardware safety requirements allocated to the AI elements	Applicable (tailoring can be necessary)	Applicable	Refinement of Hardware- Software Interface

© ISO 2024 – All rights reserved

ISO 26262:2018[a]	Interaction of the AI system with the encompassing system (the AI system is not an item)	AI system: System component consisting of hardware and software components	AI component: Conventional[b] hardware component	AI component: AI model implemented by software component(s)
Part 6: Product development at the software level	Potential source of software safety requirements allocated to the AI elements	Applicable (tailoring can be necessary)	Refinement of Hardware- Software Interface	Applicable (tailoring can be necessary)
Part 7: Production, operation, service and decommissioning	AI elements can be part of the production process	Potential source of requirements and work products relevant for production, operation, service and decommissioning	Potential source of requirements and work products relevant for production, operation, service and decommissioning	Potential source of requirements and work products relevant for production, operation, service and decommissioning

[a] n-x means ISO 26262-n:2018, Clause x

[b] conventional hardware is hardware which is not specifically designed to implement an AI model, e.g. CPUs, GPUs or FPGAs.

Table 6–2 — Example interactionswith ISO 21448

ISO 21448:2022, Clause	Interaction of the AI system development with encompassing system activities, motivated by ISO 21448:2022
5 Specification and design	Clause 5 activities: — Provide the interfaces of the AI system with the encompassing system. — Determine the semantic input space — Provide the functionality required from the AI system — Provide safety requirements allocated to the AI system, including, but not limited to, safety-related KPIs — etc ISO PAS 8800 activities: — Provide triggering conditions and functional insufficiencies of the AI system — Provide achieved safety-related KPIs — Provide a description of deployment measures required to support the AI and data lifecycles — etc
6 Identification and evaluation of hazards	Clause 6 activities are a potential source of safety requirements, including, but not limited to, safety-related KPIs, allocated to the AI system

© ISO 2024 – All rights reserved

ISO 21448:2022, Clause	**Interaction of the AI system development with encompassing system activities, motivated by ISO 21448:2022**
7 Identification and evaluation of potential functional insufficiencies and potential triggering conditions	Clause 7 activities: — Are a potential source of safety requirements allocated to the AI system — Are a potential source of potential triggering conditions of the AI system ISO PAS 8800 activities: — Provide potential triggering conditions of the AI system
8 Functional modifications addressing SOTIF-related risks	ISO PAS 8800 activities: Modification request to the encompassing system in case safety requirements allocated to the AI system cannot be fulfilled
9 Definition of the verification and validation strategy	Clause 9 activities are a potential source of safety-related KPIs, allocated to the AI system
10 Evaluation of known scenarios	ISO PAS 8800 activities — provide triggering conditions of the AI system and the associated AI error modes, error patterns or a combination of both; — Provides achieved safety-related KPIs
11 Evaluation of unknown scenarios	ISO PAS 8800 activities provideachieved safety-related KPIs
12 Evaluation of the achievement of the SOTIF	ISO PAS 8800 activities support with the safety assurance case the AI system part of the evaluation of the achievement of the SOTIF
13 Operation phase activities	ISO PAS 8800 activities provide the AI system requirements to the encompassing system regarding the operation phase

During development and as part of continuous assurance activities during operation, it can become necessary to adjust the safety requirements allocated to the AI system leading to an iterative feedback cycle to the encompassing system safety concept and safety requirements. Iterations of the requirements aretriggered for example if:

— an AI system capable of fulfilling its assigned safety requirements and associated safety-related properties cannot be feasibly developed (e.g. due to inherent performance limitations in the machine learning algorithm used);

— suitable training data and test data cannot be found; or

— evidence to demonstrate that the safety requirements and associated safety-related properties are fulfilled cannot be collected with sufficient confidence.

In each of these cases, changes to the encompassing system safety concept can be defined, leading to a set of updated, realisable requirements on the AI system.

NOTE 3 Measures on the encompassing system safety concept to reduce the safety load of an AI system towards better feasibility can be (see also 10.5):

© ISO 2024 – All rights reserved

— restrictions in the ODD;

— implementation of diversity such as different processing algorithms or sensing modalities;

— implementation of redundancy such as multiple hardware components in parallel; or

— a combination of the aforementioned measures

Once an adequate level of performance of the AI system has been achieved (in relation to the safety requirements), then the AI system can be provided for integration into the encompassing system including evidence to support the achievement of the safety requirements. This can result in the need for further iterations of the AI safety life cycle, for example, due to the following conditions:

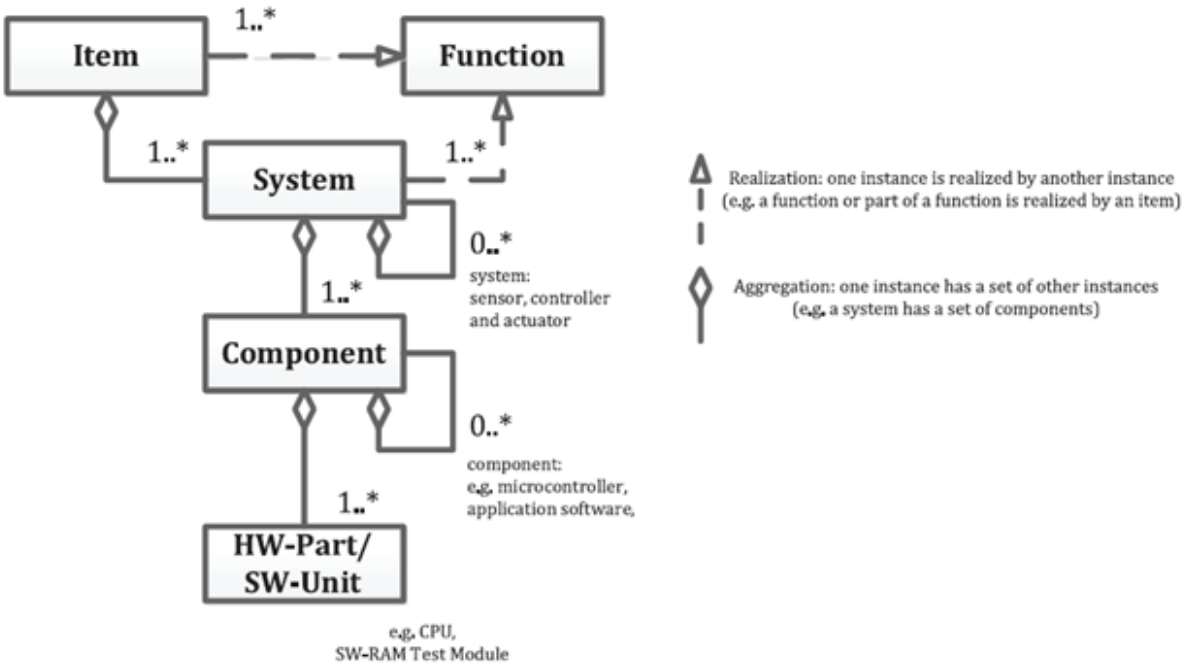
— Integration tests of the encompassing system reveal previously undiscovered faults or functional insufficiencies in the AI system that require additional development cycles.

— The encompassing system assurance case requires additional evidence to support safety claims related to the AI system that require additional effort to collect the required evidence.

Collected field data and observations made during operation related to the performance of the AI system (e.g. increased number of false positive errors under certain traffic conditions or an increased rate of out-of-distribution inputs) can indicate changes in the input space. Those changes might not be able to be addressed by a refinement of the safety requirements on the AI system or through additional development activities but require changes to be made at the encompassing system level. In turn, this can lead to changes in the safety requirements assigned to the AI system and a repetition of the safety life cycle.

6.3 Mapping of abstraction layers between ISO 26262, ISO/IEC 22989 and this document

ISO 26262:2018 uses the following levels of abstraction: item, system, component, software unit and hardware part. The relationship of these are visualized in Figure 6–2 which corresponds to ISO 26262-10:2018, Figure 3. The term “element” can mean “system”, “component”, “software unit” or “hardware part”, depending on the context. It is typically used when a given requirement can be applied on different levels of abstraction, e.g. on hardware component as well as on hardware part level. An example item composition is shown in Figure 6–3 which corresponds to ISO 26262-10:2018, Figure 4.



© ISO 2024 – All rights reserved

NOTE 1 Depending on the context, the term “element” can apply to the entities “system”, “component”, “hardware part” and “software unit” in this chart, according to ISO 26262-1:2018, 3.41.

NOTE 2 "*" sign means N elements are possible, where N is a positive integer number.

Figure 6–2 — Relationship of item, system, component, hardware part and software unit

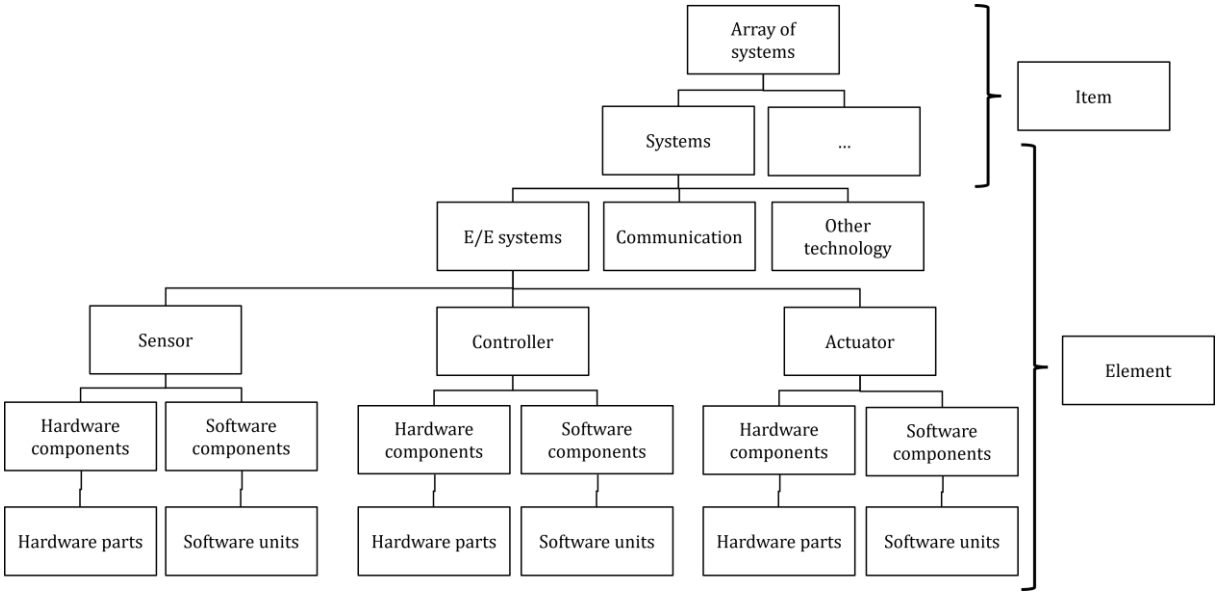
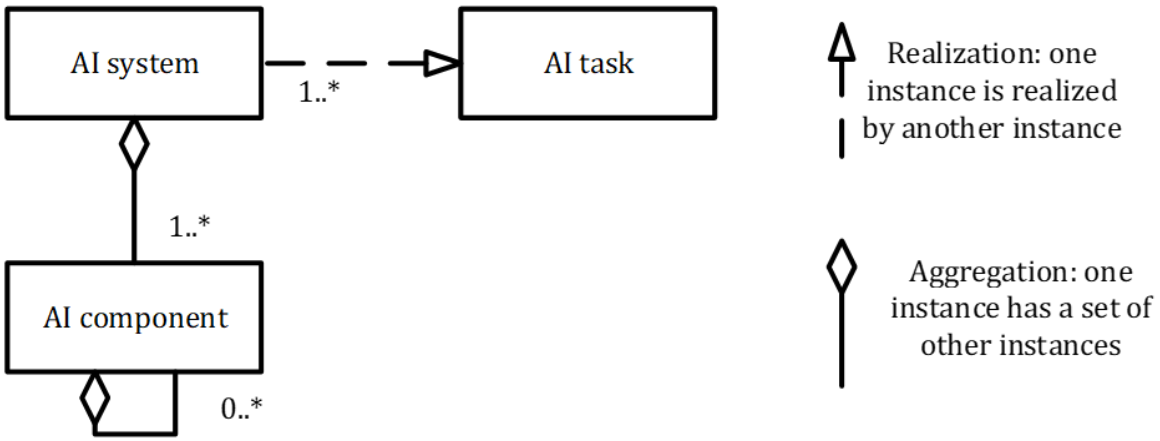


Figure 6–3 — Example item composition

ISO/IEC 22989 uses the following abstraction layers: AI system and AI components, where the AI system consists of AI components. ISO/IEC 22989 does not explicitly state if a given AI component can itself consist of AI components. In this document this is possible. The AI component can be an AI model, a conventional element, i.e. an element not considered to be an AI model, or a combination of both. The AI system contains at least one AI model and realizes the AI task. Figure 6–4 uses the same notation as Figure 6–2 to visualise the relationship between AI system and AI components.



NOTE 1 Depending on the context, the term “AI element” can apply to “AI system” and “AI component”

NOTE 2 "*" sign means N elements are possible, where N is a positive integer number.

© ISO 2024 – All rights reserved

Figure 6–4 — Relationship of AI system and AI component

This document uses terms from both ISO/IEC 22989:2022 and ISO 26262-1:2018. The terms from the different standards do not map one-to-one. So, depending on the context, multiple mappings are possible as shown in Table 6–3

Table 6–3 — Possible mappings between ISO/IEC 22989 and ISO 26262 terms, depending on the context

ISO/IEC 22989:2022 terminology	ISO 26262:2018 terminology
AI system	Item, system, component, software unit or hardware part Element
AI component	System, component, software unit or hardware part Element

The term “AI system” serves in this document as the top level of abstraction of the content to be developed. As such it is possible in a distributed development that what one party considers to be an AI component, the other party considers to be an AI system, as for them it represents the top level of the content they develop.

Figure 6–5 provides an example of an item decomposed into its elements. In this example the item consists of two systems, system A and system B. For the sake of simplicity system B is not further decomposed. System A is composed out of the system components A.1 and A.2. System component A.1 does not contain an AI model. As such it cannot be an AI system. System component A.2 contains an AI model and is declared to be the AI system in this example.

NOTE It would have also been possible to declare system A as the AI system, as it too contains at least one AI model. The decision regarding the scope of the AI system is made as part of negotiations between the development organisations responsible for the encompassing system.

The system components themselves consist of hardware and software components. In case of system component A.2 these components are considered to be AI components as they compose the AI system. A further breakdown into software units and hardware parts of the components has been omitted for sake of simplicity.

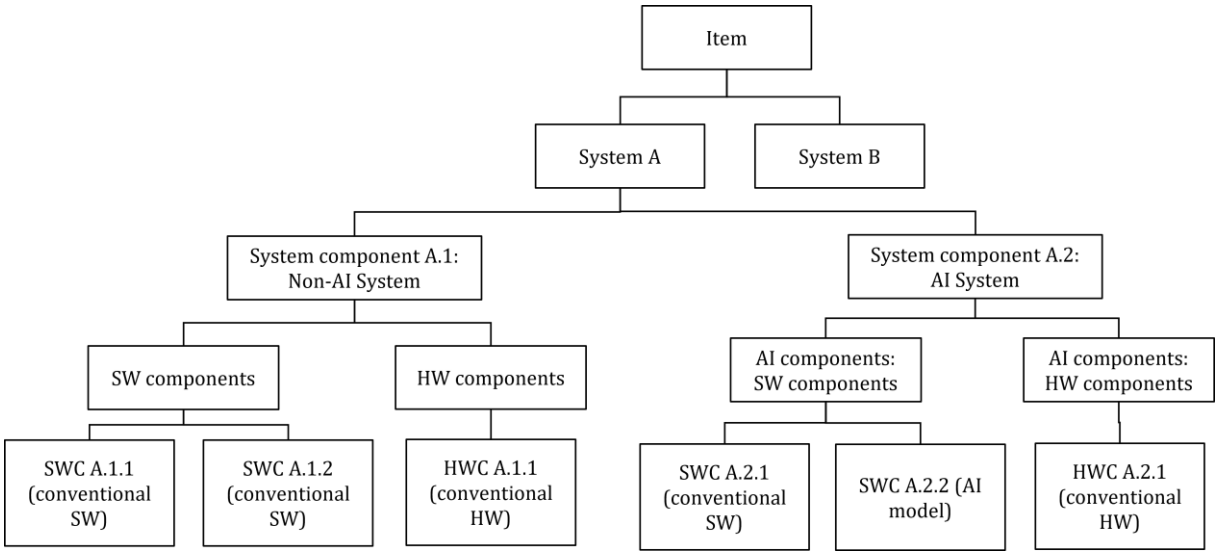


Figure 6–5 — Example of a hierarchical decomposition of an item into its elements down to the component level - decomposition tree view

© ISO 2024 – All rights reserved

6.4 Example architecture for an AI system

This document uses the architecture shown in Figure 6–6 as an example architecture. The AI system receives its input from the source, executes its task based on the input and the control signals and then provides its output to the consumer. The AI system itself consists of the AI components AI pre-processing, AI model and AI post-processing. The AI post-processing hereby uses data provided from the previous process steps (i.e. AI pre-processing and the AI model) in combination with the original input data for monitoring purposes.

NOTE 1 This architecture is just an example and has no claim of representing all possible architectures of AI systems.

NOTE 2 If the AI system is implemented for a real-time task, the execution of the AI system could be triggered synchronously or asynchronously depending on the behaviour of the control element, sources of the input streams and the consumers of the outputs. These considerations can be relevant to the definition of the safety requirements on the AI system.

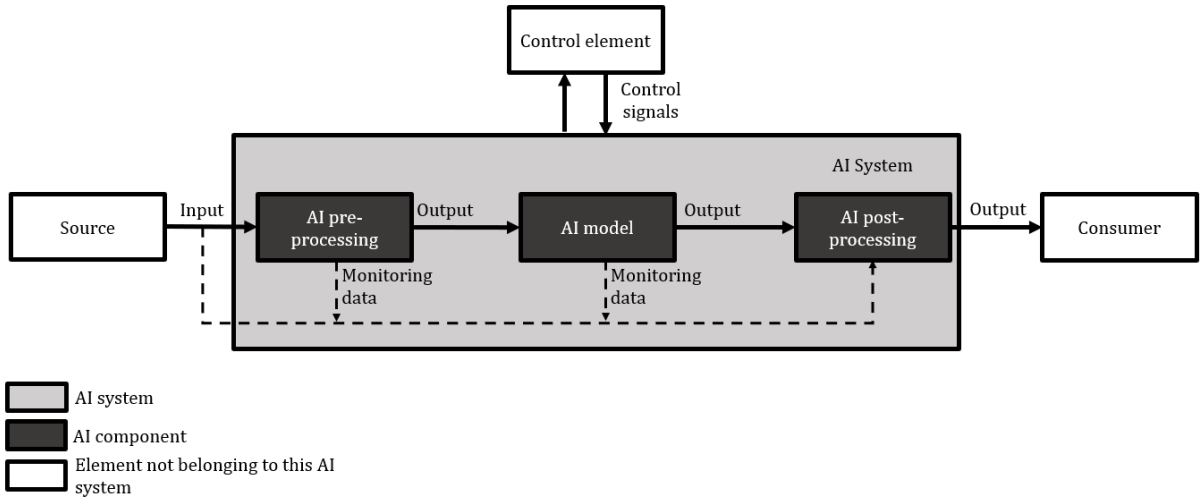


Figure 6–6 — Example architecture of an AI system

6.5 Types of AI models

Examples for types of AI models include, but are not limited to, deep neural networks, k-nearest neighbours, support vector machines, decision trees, symbolic AI and fuzzy logic. These can be clustered in different categories. For example, deep neural networks, k-nearest neighbours, support vector machines and decision trees can be categorized as ML models as shown in Figure 6–7

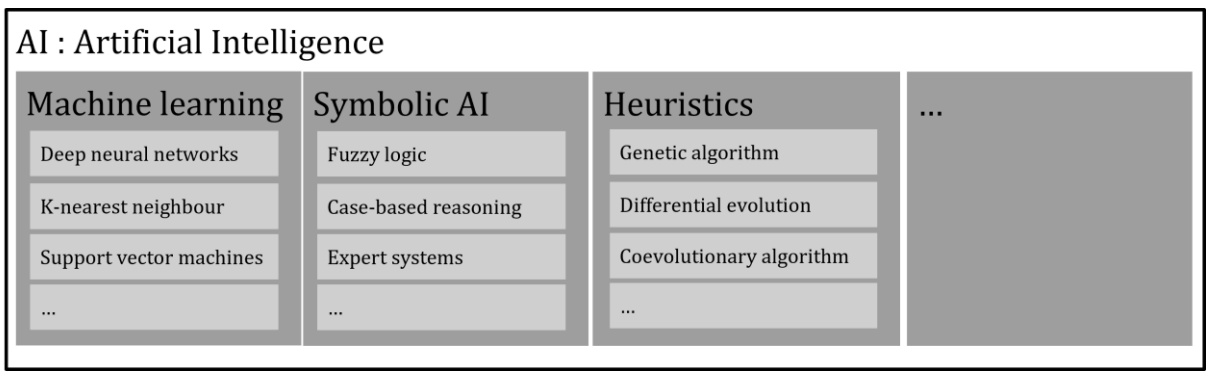


Figure 6–7 — Example of different types of AI models

© ISO 2024 – All rights reserved

6.6 AI technologies of a ML model

Figure 6–8 and Figure 6–9 show an example of possible AI technologies utilized for a ML model that is implemented in hardware and software. Next to the AI method itself the AI technology also contains the tools and procedures to generate the AI model.

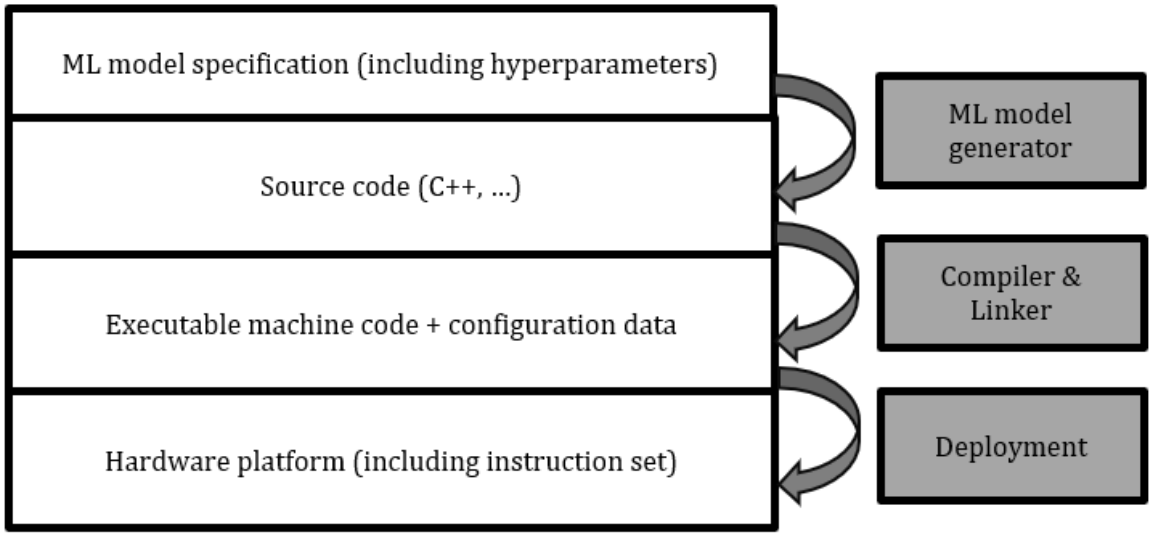
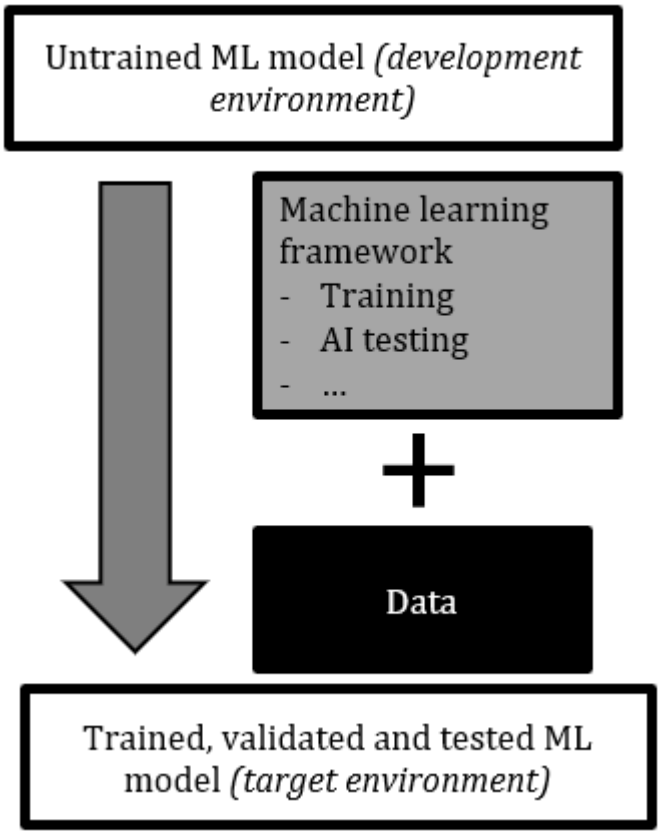


Figure 6–8 — AI technologies to create an executable ML model (application of ISO 26262)



© ISO 2024 – All rights reserved

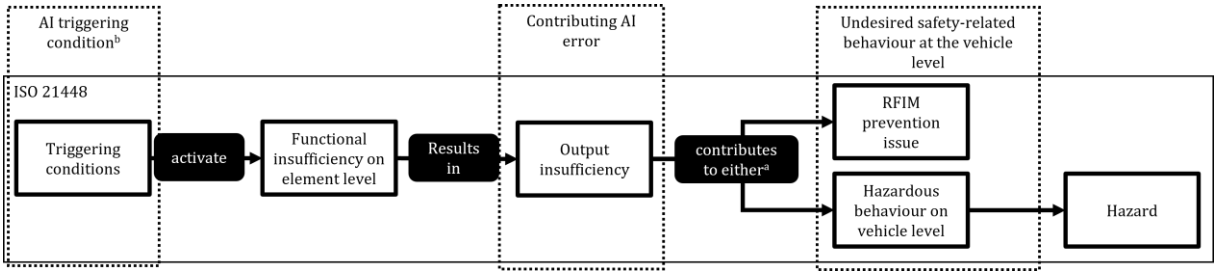
Figure 6–9 — AI technologies to create a trained ML model (application of this document)

The AI technologies listed in Figure 6–8 are not considered to be relevant sources for functional insufficiencies, e.g. compiler and linker are well known technologies already utilized by non-AI systems. For these technologies the application of the ISO 26262 series is considered to be sufficient in order to achieve AI safety. The AI technologies listed in Figure 6–9 are considered to be relevant sources of functional insufficiencies for which the application of the ISO 26262 series alone is not considered to be sufficient to achieve AI safety. For these technologies the remaining clauses of this document are applied.

6.7 Error concepts, fault models and causal models

6.7.1 Cause-and-effect chain

This document utilizes the concept of AI triggering conditions, faults, functional insufficiencies, AI errors and the undesired safety-related behaviour at the vehicle level. The mapping of the terms of this document to the cause-and-effect chain used by ISO 21448:2022 can be found in Figure 6–10. The mapping of the terms of this document to the cause-and-effect chain used by ISO 21448:2022 can be found in Figure 6–11. The AI triggering condition activates a fault or a functional insufficiency, resulting in an AI error in case of a fault and a contributing AI error in case of a functional insufficiency.



Key

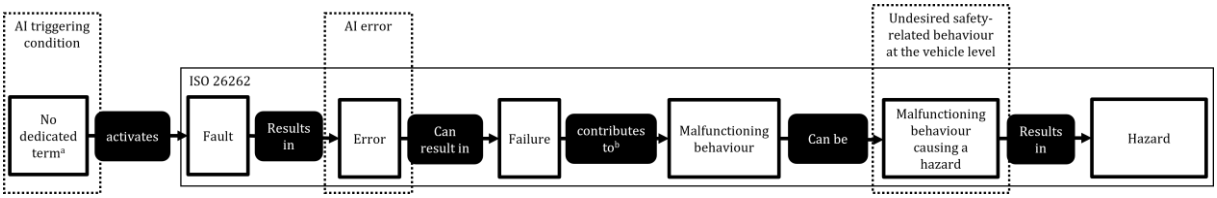
a An output insufficiency, either by itself or in combination with one or more output insufficiencies of other elements, contributes to either a hazardous behaviour at the vehicle level or an inability to prevent or detect and mitigate a reasonably foreseeable indirect misuse.

b Since a triggering condition of ISO 21448 results in a contributing AI error in the context of this document, they represent a subset of all AI triggering conditions

Figure 6–10 — Mapping of the cause-and-effect chain of ISO 21448 to the terms of this document

EXAMPLE An insufficiency of specification could be a missing object in the AI training, AI validation and AI test dataset of an AI system utilizing an ML model for object classification. In this case encountering this object during operation in the field is the AI triggering condition activating this insufficiency of specification, resulting in the occurrence of a contributing error. The contributing AI error would be the incorrect classification of the object by the ML model and consequently by the AI system.

An AI error of an AI component can propagate through the AI system and can result in an AI error of the AI system. The AI error of the AI system can propagate through the encompassing system and can contribute either by itself or in combination with one or more other errors or output insufficiencies of the elements of the encompassing system to an undesired safety-related behaviour at the vehicle level.



© ISO 2024 – All rights reserved

Key

a In ISO 26262 there is neither a dedicated term for the condition which activates a fault nor is this concept explicitly utilized.

b A failure, either by itself or in combination with one or more failures of other elements, contributes to a malfunctioning behaviour.

Figure 6–11 — Mapping of the cause-and-effect chain of ISO 26262 to the terms of this document

The undesired safety-related behaviour at the vehicle level is used as an umbrella term for the corresponding terms of ISO 26262:2018 (i.e. the malfunctioning behaviour at the vehicle level which can cause hazards) and ISO 21448:2022 (i.e. the hazardous behaviour at the vehicle level and RFIM prevention issue).

6.7.2 Root cause classes

Different error classes can be distinguished depending on the root cause. The correlation of contributing AI errors and their different root causes is shown in Figure 6–12.

NOTE 1 A contributing AI error of the AI element can lead to the undesired safety-related behaviour at the vehicle level by itself or in combination with one or more other AI errors.

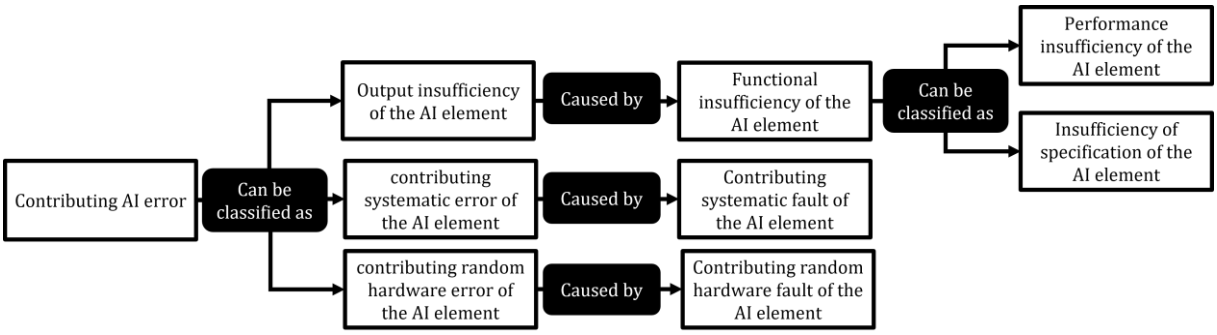


Figure 6–12 — Correlation of safety-related errors with their different classes of root causes

The root causes for the different kinds of AI errors are:

— Insufficiency of the specification

EXAMPLE 1 An insufficiency of the specification could be missing datasets in the AI training, AI validation or AI test dataset. The resulting output insufficiency could be a misclassification when exposed to the missing datasets.

EXAMPLE 2 Specification of a neural network model with insufficient complexity.

EXAMPLE 3 An inadequate training loss function.

EXAMPLE 4 Inadequate labelling specification

— Performance insufficiency

EXAMPLE 5 A performance insufficiency could be an insufficient range of a sensor in case of certain environmental conditions. The resulting output insufficiency could be a false negative detection of an obstacle in the trajectory.

NOTE 2 In the case of ML models, performance insufficiencies can be specifically caused by training and test dataset related issues, e.g. insufficiencies in the coverage of the respective input space. These data-related issues are in turn considered to be insufficiencies of specification, or more precisely as insufficiency of specification of the data.

— Contributing systematic fault

© ISO 2024 – All rights reserved

EXAMPLE 6 A contributing systematic fault could be to divide by zero in software or to use incorrect variable names.

EXAMPLE 7 Overfitting the DNN resulting in wrong high-confidence classification outputs of corner cases can be regarded to be a contributing systematic fault in the training procedure.

NOTE 3 Sometimes the classification of a given issue in either a systematic fault or a functional insufficiency can be ambiguous. Independent of the classification a safety assurance argument is provided to argue that this issue does not represent an unreasonable risk. As long as this safety assurance argument is available, the exact classification is not relevant.

— Contributing random hardware fault

EXAMPLE 8 Physical defect causing a short to ground.

When evaluating the effectiveness of safety mechanisms, it can be necessary to distinguish the different classes of errors.

EXAMPLE 9 Homogenous redundancy can be effective in detecting random hardware errors in one of the redundant elements, but it is not effective in detecting systematic errors.

The different fault classes can also be used within the safety assurance argument by addressing each root cause class with a dedicated set of safety measures.

EXAMPLE 10 Coding guidelines are a measure to avoid systematic faults in SW. In combination with other fault avoidance measures, e.g. as specified in ISO 26262-6:2018, the absence of unreasonable risk due to systematic faults could be argued.

It can also be useful to classify the AI triggering conditions in different categories.

EXAMPLE 11 For ML based AI models the AI triggering conditions can be distinguished as:

— cases that are similar to known training data samples ("in-distribution" cases), such as: cases at decision boundaries ("hard" cases), and undecided cases (high aleatoric uncertainty e.g. due to label noise);

— unseen cases ("out-of-distribution" cases), such as: novel objects (semantic shift), or novel image styles (covariate shift);

— addition of small changes to a sample which causes no error (e.g. addition of adversarially crafted perturbation)

— applying a maliciously inserted trigger pattern (e.g. inserted via data or model poisoning)

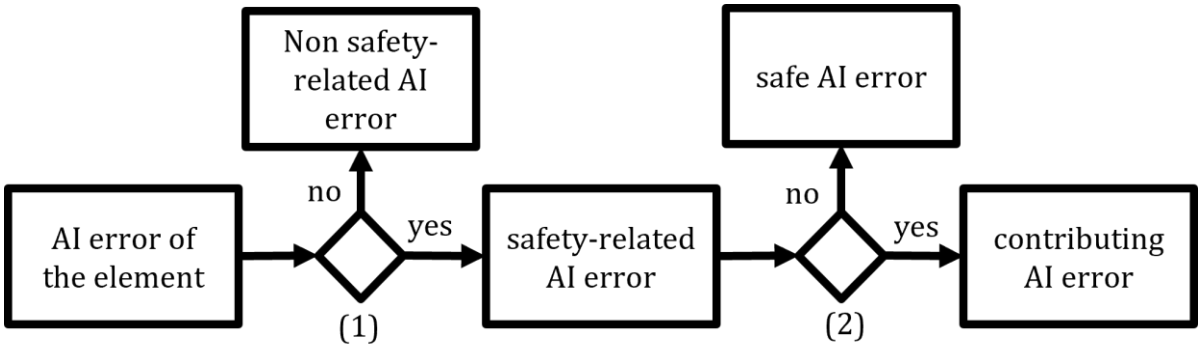
Statistically obtained ML models like deep neural networks exhibit for any output an inherent uncertainty about their correctness. Therefore, besides classification of AI errors, root causes, and AI triggering conditions, also a classification of uncertainty types and their associated sources can be helpful in practice.

EXAMPLE 12 The two major types used to model the uncertainty of the ML model are epistemic and aleatoric uncertainty. While epistemic uncertainty stems from uncertainty of having the right model for the given sample and can often be fixed with more data, aleatoric uncertainty stems from intrinsic noise in the training data.

6.7.3 Error classification based on the safety impact

Compared to the criticality classification of random hardware faults as described in ISO 26262-5:2018 this document uses a simplified scheme as shown in Figure 6–13.

© ISO 2024 – All rights reserved



Key

(1) Is the element a safety-related element?

(2) Can the error significantly contribute to the occurrence of a safety-related undesired behaviour at the vehicle level?

Figure 6–13 — Error classification scheme based on the potential to lead to an undesired safety-related behaviour at the vehicle level

7 AI safety management

7.1 Objectives

The objectives of this clause are:

a) to define an AI safety lifecycle and its activities to ensure that contributing errors of the AI system do not lead to unreasonable risk of undesired safety-related behaviour at the vehicle-level. The AI safety lifecycle includes:

   1) a definition of activities necessary to develop the AI system, to provide the assurance and the evidence that the AI system is safe and to ensure the AI safety during operation;

   2) in case of the utilization of ML based AI techniques: a data-driven, iterative approach for the development, evaluation and continuous assurance of AI-based functions within the context of a system-level safety lifecycle;

b) to ensure that overall and project specific safety management processes and activities are appropriate to ensure the safety of the AI system;

   NOTE ISO 26262-2:2018 provides suitable guidance on overall safety management and project specific safety management. This guidance can require extensions based on recommendations in this document.

c) to plan, initiate and conduct the AI safety activities.

7.2 Prerequisites and supporting information

The following information shall be available (from external sources):

a) the AI system definition (from external sources), including:

   1) the AI system functionality;

   2) the interfaces of the AI system with the encompassing system, including if applicable, the ASIL capability of the inputs to the AI system.

© ISO 2024 – All rights reserved

3) the safety requirements allocated to the AI system, including if applicable:

i) the ASIL rating of the safety requirements;

ii) the acceptance criteria or validation targets derived in compliance with ISO 21448:2022, Clause 6 or 9.

NOTE 1 The safetyrequirements allocated to the AI system from external sources are typically requirements regarding the avoidance or control of safety-related faults, the allowed maximum error occurrence rate of contributing AI errors, the identification of AI triggering conditions, and the robustness against certain environmental conditions.

NOTE 2 For an elaboration of the fault model, the causal model, and the error concepts used by this document, see 6.7.

7.3 General requirements

7.3.1 An AI safety lifecycle shall be defined that specifies the activities necessary to develop the AI system, to provide the assurance and the evidence that the AI system is safe and to ensure the maintenance of AI safety during operation. It can be based on the reference lifecycle (Figure 7–2) and can be tailored according to project specific needs. The tailoring is supported by a rational why the tailored AI safety lifecycle is sufficient and appropriate to achieve AI safety.

7.3.2 At each phase within the AI safety life cycle, work products shall be defined to support the safety assurance claims of the AI system.

7.3.3 The activities of the AI safety lifecycle shall be coordinated with the safety lifecycle activities of the encompassing system as defined by ISO 26262-2 and, if applicable, ISO 21448.

NOTE The AI system can be developed as a safety element out ofcontext. The concept of a safety element out of context is described in ISO26262-10:2018, Clause 9 "Safety Element out of Context".

7.3.4 The activities described in ISO 26262-2:2018 shall be adapted in order to address the management of AI safety, including:

a) theintegration of the AI safety lifecycle into the ISO 26262 safety lifecycle (see Figure.2 of ISO26262-2:2018);

b) the enhancement of “functional safety” to “AI safety”;

c) measures to ensure that a sufficient level of cross domain competences regarding safety and AI are available, in compliance with ISO 26262-2:2018, 5.4.4.1;

d) adding this document as a relevant standard of ISO 26262-2:2018;

e) the use of a safety assurance argument as part of the safety case of ISO 26262-2:2018;

f) extending the safety plan of ISO 26262-2:2018 to includethe safety activities of this document;

g) Tailoring of Table 1 of ISO 26262-2:2018 to address the workproducts of this document (see Table 7–1).

Table 7–1 — Required confirmation measures, including the required level of independence

© ISO 2024 – All rights reserved

Confirmation measure	Level of independence[a] applies to					Scope
	QM	ASIL A	ASIL B	ASIL C	ASIL D	
Confirmation review of the safety plan Independence with regard to the developers of the item[b], project management and the authors of the work product	-	I1	I1	I2	I3	Applies to the highest ASIL among the safety requirements
Confirmation review of the AI system validation report Independence with regard to the developers of the item[b], project management and the authors of the work product	-	I0	I1	I2	I2	Applies to the highest ASIL among the safety requirements
Confirmation review of the AI safety analyses Independence with regard to the developers of the item[b], project management and the authors of the work product	-	I1	I1	I2	I3	Applies to the highest ASIL among the safety requirements
Confirmation review of the safety assurance argument Independence with regard to the authors of the safety	-	I1	I1	I2	I3	Applies to the highest ASIL among the safety requirements

© ISO 2024 – All rights reserved

Confirmation measure	Level of independence[a] applies to					Scope
	QM	ASIL A	ASIL B	ASIL C	ASIL D	
assurance argument						
AI safety audit Independence with regard to the developers of the item[b] and project management	-	-	I0	I2	I3	Applies to the highest ASIL among the safety requirements
AI safety assessment Independence with regard to the developers of the item[b] and project management	-	-	I0	I2	I3	Applies to the highest ASIL among the safety requirements

[a] The notations are defined as follows:

— -: no requirement and no recommendation for or against regarding this confirmation measure;

— I0: the confirmation measure should be performed; if the confirmation measure is performed, it shall be performed by a different person in relation to the person(s) responsible for the creation of the considered work product(s);

— I1: the confirmation measure shall be performed, by a different person in relation to the person(s) responsible for the creation of the considered work product(s);

— I2: the confirmation measure shall be performed, by a person who is independent from the team that is responsible for the creation of the considered work product(s), i.e. by a person not reporting to the same direct superior;

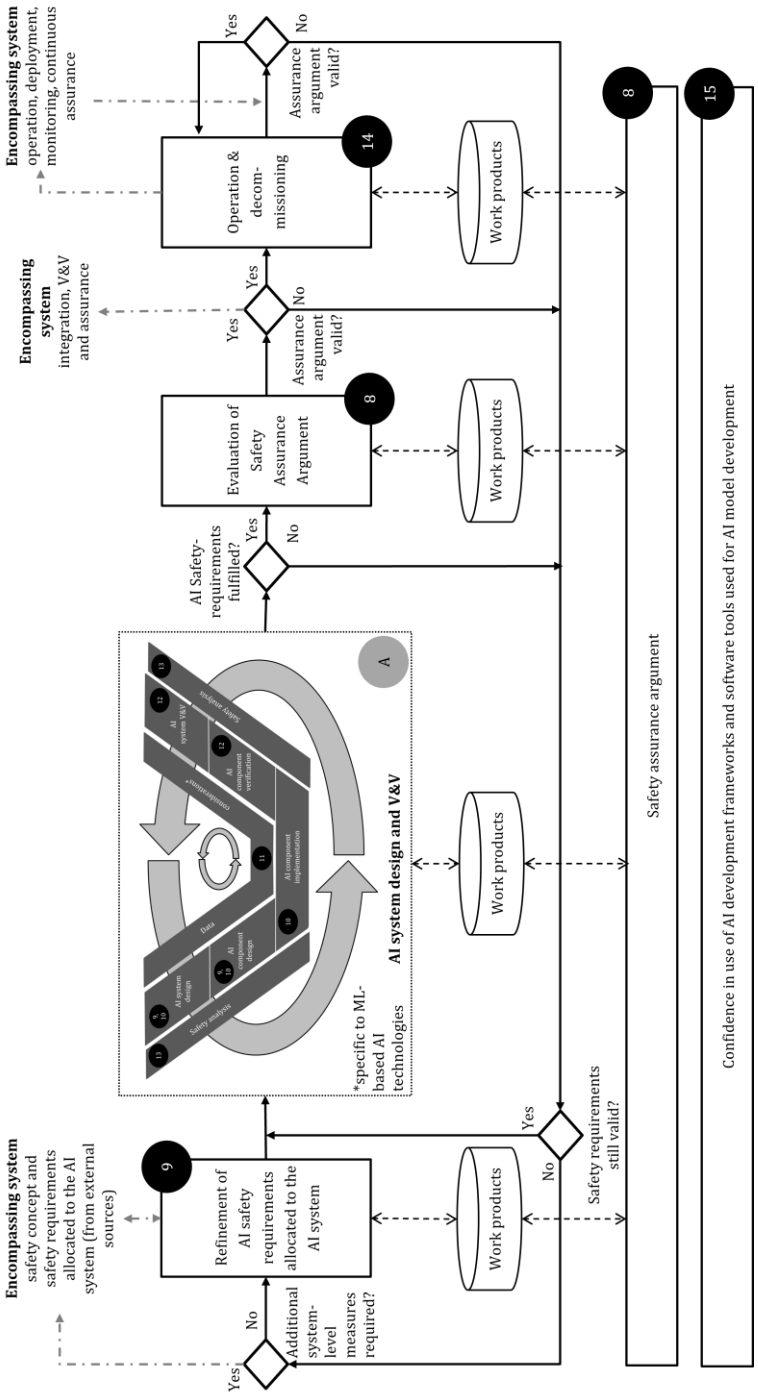
— I3: the confirmation measure shall be performed by a person who is independent, regarding management, resources and release authority, from the department responsible for the creation of the considered work product(s).

[b] The developers of the item include the developers of the AI system

7.4 Reference AI safety life cycle

The reference AI safety life cycle described in this clause covers the activities at the different phases of the AI system development, deployment and operation: safety-related requirements derivation, AI system design, verification and validation, deployment and operation. The AI safety life cycle is summarised in Figure 7–1and is used to structure the remainder of this document. A detailedview of the AI system design and V&V phase is shown in Figure 7–2. Clause 8 through Clause 15(as indicated by the numbered black dots in Figure 7–1 and Figure 7–2) are used to describe the activities within the safety life cycle in more detail.

© ISO 2024 – All rights reserved

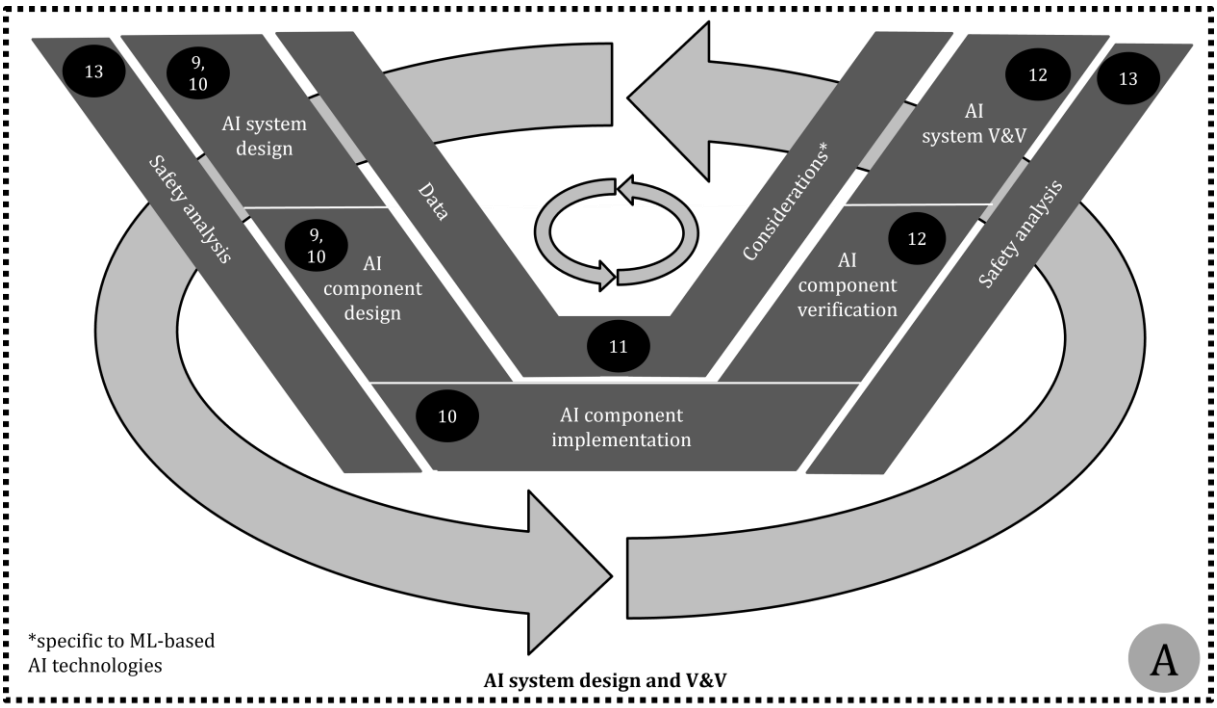


Key

black circle with numbers	represent clause number(s)
arrow	show the flow and iterative nature in particular of the AI component design, implementation and verification and the whole AI system design and verification

Figure 7–1 — Reference AI safety lifecycle

© ISO 2024 – All rights reserved



Key

black circle with number(s) — represent clause number(s)

arrow — show the flow and iterative nature in particular of the AI component design, implementation and verification and the whole AI system design and verification

Figure 7–2 — Detailed view of the AI system design and V&V phase of the reference AI safety lifecycle

7.5 Iterative development paradigms for AI systems

The AI system development phase of the AI safety life cycle covers all activities required to design, implement, verify and validate the AI system.

The AI system development activities described in Clause 8through Clause 13 are iteratively performed until a sufficient level of performance with respect to the safety requirements and the associated safety properties can be demonstrated and associated workproducts are generated.

An essential characteristic of this development process is the analysis to identify potential functional insufficiencies, their root causes,and their impact on safety. This analysis is used to derive appropriate measures to reduce these functional insufficiencies during design (including the selection of training data in case of ML) as well as to reduce the impact of contributing errors (through architectural measures).

NOTE 1 During the AI system development, the property of "independence" is considered wherever appropriate (e.g. independence between training datasets and test datasets). An analysis similar to the Dependent Failure Analysis (DFA) described in ISO 26262-9:2018, Clause 7 can be used.

The development model described here reflects the iterative development model typically used in the area of machine learning with a focus on activities to identify, analyse, reduce and mitigate functional insufficiencies in the trained model and the cumulative collection of evidences to support claims regarding safety requirements allocated to the function. Furthermore, for machine learning, the specification and collection of suitable training and test data is one of the most influential factors for the performance of the function. Therefore, the specification, planning, collection, acquisition, preparation and the labelling of data related to

© ISO 2024 – All rights reserved

AI component implementation and verification process is treated in this document as asafety-related development activity with specific objectives and associated safety artefacts (see Clause 11).

The iterative development of the AI system is guided by a set of performance indicators, including safety-related properties associated with the safety requirements allocated to the AI system.

NOTE 2 "Iteration" in the context of AI system development can be defined as a single repetition of one or more AI system safety lifecycle phases, that is applied until a set of conditions defined by a set of KPIs or other target parameters are either fulfilled or demonstrated to be unachievable.

NOTE 3 Considering that there can be multiple, potentially conflicting target KPIs, the presence of contributing errors in the AI system can be inevitable.

Despite the inherently iterative nature of the AI system development, different iteration cycles of the development can focus on different sets of performance indicators, depending on the maturity of the development.

Examples of such distinct iteration cycles can include:

— **Proof of concept:** In this phase of development, AI models aredesigned, implemented and tested against a set of initially defined safety requirements. The objective of this phase is to evaluate the potential of the chosen AI technologies to fulfil the safety requirements, as well as defining a set of measures required to minimise the number or the impact of functional insufficiencies in the function (e.g. optimisation of model parameters and defining adata collection strategy).

— **Series development of the AI system:** In this phase of development, the AI system is iteratively developed against all safety requirements. Errors of the AI system or the AI model are analysed with respect to their potential root causes and measures are defined to reducethe functional insufficiencies through design and data selection or for minimising their impact through architectural measures. This phase of development can use a host platform for implementing and testing the AI model and will continue until an adequate level of performance for all safety-related properties has been met.

— **Deployment to the target hardware platform:** In this phase of development, the AI system is transferred to the target hardware platform and target software platform. This can include, for example a change in numerical precision used to calculate the results as well as the consideration of constraints such as timing, memory limitations (e.g. resulting in the necessity to prune the computational graph) and robustness against potential random hardware faults. The focus of this development phase is to ensure that the safety requirements are nevertheless met, despiteany limitations of the target hardware platform and the target software platform.

— **Furtherimprovement after deployment:** In this phase of development, the AI system is iteratively updated based on observations made during operation in the field and new requirements mandated by the developer. This can include compensating for previously unknown triggering conditions (e.g. concept drift) and distributional shift in the environment (e.g. domain shift). Performance indicators within this phase of development are monitored to ensure a monotonic safety improvement with respect to previous iterations of the AI model. This phase of development can also include development versions of the function running in "shadow" mode within an operational environment in order to collect suitable data and evaluate the potential performance under realistic conditions.

7.6 Work products

7.6.1 AI safety lifecycle resulting from 7.3.1 to7.3.4.

7.6.2 Work products of ISO 26262-2:2018, 5.5, resulting from 7.3.4.

7.6.3 Work products of ISO 26262-2:2018, 6.5,resulting from 7.3.3and 7.3.4, in particular the safety plan.

© ISO 2024 – All rights reserved

7.6.4 Work products of ISO 26262-2:2018, 7.5, resulting from 7.3.4.

8 Assurance arguments for AI systems

8.1 Objectives

The objectives of this clauseare:

a) to develop an assurance argument demonstrating that the safety requirements allocated to the AI system are fulfilled;

NOTE 1 The assurance argument for the AI system contributes to the safety assurance argument of the encompassing system.

NOTE 2 The assurance argument can be developed independently from the encompassing system as a safety element out of context (SEooC) development activity (see ISO 26262-10:2018, Clause 9[5]). In such cases, the assurance argument documents all necessary assumptions on the encompassing system for arguing that the safety requirements allocated to the AI system are fulfilled.

b) to evaluate whether the assurance argument reflects the actual residual risk of the AI system violating its safety requirements.

8.2 Prerequisites and supporting information

The following information shall be available at the initiation of these activities:

a) the AI system definition (from external sources), including:

1) a specification of the safety requirements allocated to the AI system;

2) a definition of the technical context within the encompassing system (e.g. definition of interfaces to and from the encompassing system and, if applicable, the environment, conditions under which the AI system functionality is triggered, etc.);

NOTE 1 This includes the ASIL capability and noise to signal ratio of the input signals provided by the source, if applicable.

3) a specification of the input space;

b) requirements on the assurance argument and work products for the AI system (from external sources). These requirements can be derived from the assurance argument of the encompassing system as well as safety management procedures from Clause 7;

The following information shall be available for the finalization of these activities:

c) the work products of the AI safety lifecycle;

NOTE 2 This body of evidence can be cumulatively collected as part of the iterative development process phases or produced within dedicated development process cycles

The following information can be considered for the finalization of this phase:

d) required properties of the encompassing system to achieve AI safety;

e) relevant properties of the input space (e.g. distribution of critical events, physical constraints on changes of input values over time);

© ISO 2024 – All rights reserved

f) evidence of organization-specific rules and processes for AI safety, evidence of competence management and evidence of a quality management system, from 6.6.2;

g) evidence that the organization-specific rules and processes have been followed and that the work products have the required maturity and quality, from 6.6.3 and 6.6.4.

8.3 General requirements

8.3.1 An assurance argument for the fulfilment of the safety requirements allocated to the AI system shall be provided.

NOTE 1 The assurance argument can be part of the encompassing system's safety case in accordance with ISO 26262-2:2018, 6.4.8.

NOTE 2 The assurance argument can be constructed at the level of the encompassing system. In this case, the development of the AI system-specific contributions and supporting evidence are considered part of the AI safety life cycle and therefore within the scope of this document.

8.3.2 The assurance argument shall usethe relevant work products generated during the AI safety lifecycle to support the assurance claims.

NOTE Changes to the work products and their impact on the assurance argument are considered as part of change management throughout the AI safety life cycle.

8.3.3 Confirmation measures of 7.3.4 shall be applied to the assurance argument.

NOTE The evaluation of the validity of the assurance argument can be performed as part of confirmation measures of the encompassing system (see ISO 26262-2:2018, 6.4.9 and ISO 21448:2022, 12.3), or as an independent activity, for example in the case of a SEooC (see ISO 26262-10:2018, clause 9) or as part of a distributed development.

8.4 AI system-specific considerations in assurance arguments

A number of AI system-specific considerations impact the creation of the assurance argument.

a) The formulation of the AI safety requirements (see Clause 9):

- These include quantitative properties expressed in the form of probabilities (e.g. proportion of false positive classifications).
- Arguments are expressed that demonstrate that these properties have been achieved with a level of statistical confidence appropriate to the quantitative targets associated with the requirement's acceptance criteria.
- This can lead to additional requirements on the nature of evidence to support the claim and how the validity of this evidence is evaluated.

b) Statistical arguments related to aggregated performance metrics:

- These might not be sufficient to argue a suitable level of safety in rare, but critical situations (e.g. edge cases, sensor defects or adversarial perturbations).
- Arguments can be required to demonstrate that such input conditions nevertheless lead to a suitable level of AI safety.
- The probability of unknown triggering conditions leading to a violation of safety requirements can depend on:

© ISO 2024 – All rights reserved

— features of the input space not directly related to the function (e.g. due to spurious correlations in the training data) and

— predictions based on past inputs (e.g. the accuracy of previous detections of dynamic objects could impact the future behaviour of a planning task).

c) Verification of the AI system:

— Direct introspective approaches of AI models might not be effective.

— Alternative means of arguing the correct behaviour of the AI model or an increased reliance on indirect verification (e.g. test) can be required.

EXAMPLE Due to the lack of transparency with respect to the individual contributions of the large number of parameters used in some machine learning models, introspective approaches to verification might have limited applicability.

d) Reliance on training and test data:

— In machine learning-based AI methods, the behaviour of the AI system as well as its verification and validation are predominantly reliant on the selection of suitable training and test datasets as well as the training procedures themselves.

— Dedicated assurance arguments for demonstrating how the data selection process supports the achievement of AI safety might be required (see Clause 11).

— These arguments consider the training process and associated tools (see Clause 15).

e) Conditions during operation:

— Conditions can occur during operation that invalidate the assurance argument due to the complex nature of the environment in which vehicles containing AI systems are deployed.

— These conditions might include distributional shift of the input space (e.g. new types of road vehicles, changes in road infrastructure), changes to the technical system (e.g. replacement or upgrade of sensors) or previously undiscovered unknown triggering conditions.

— A continual, periodic re-evaluation and adaptation of the assurance argument is therefore performed, including an impact analysis of which parts of the assurance argument and associated evidence are to be re-evaluated (see Clause 14).

NOTE The degree to which the continual, periodic re-evaluation is required depends on properties of the input space and operating environment. If the input space, its distribution, and change of distribution over time (e.g. due to ageing) are well known, re-evaluation can be performed within regular software update activities.

8.5 Structuring assurance arguments for AI systems

8.5.1 Context of the assurance argument

Within the scope of this document, assurance relates to the claim that the AI system achieves AI safety. An assurance argument communicates the relationship between evidence and the AI safety requirements.

The level of confidence in the assurance argument should be appropriate to the required level of integrity (functional safety) and acceptance criteria assigned to the AI system within the context of the encompassing system.

© ISO 2024 – All rights reserved

NOTE A model-based graphical representation of the assurance argument can aid the communication and evaluation of the assurance argument. Examples of graphical notations for assurance arguments include the Goal Structuring Notation (GSN)[6] and Claims Argument Evidence (CAE)[7] based on the Structured Assurance Case Metamodel (SACM)[8].

The structure of the assurance argument can appeal to:

— features of the implemented item (product argument); or

— features of the development measures and assessment process (process argument); or

— factors impacting the residual risk associated with the AI system (e.g. potential causes of insufficiencies and failure modes).

The assurance argument can also include a combination of the above perspectives.

EXAMPLE 1 Process focused aspects of the assurance argument for the AI system can include an argument for the appropriate tailoring of the AI safety life cycle and the effectiveness with which each activity has been performed, based on an evaluation of the work products developed in each phase.

EXAMPLE 2 A risk-oriented assurance structure can include an argument that all possible causes of contributing AI errors are identified (e.g. via safety analyses), and suitable countermeasures for each cause have been identified and implemented, either through specific development measures or dedicated architectural measures.

The assurance argument for the AI system begins with a claim that the safety requirements allocated to the AI system are achieved. This can include statements related to a reasonable level of residual AI errors with respect to the AI safety requirements and the target functionality of the AI system.

An explicit definition of the context, as well as relevant assumptions, increases the transparency of the assurance argument and limits the scope of the argument to the specific AI system, its technical context within the encompassing system and its operating conditions.

EXAMPLE 3 Examples of context information that can be referenced by the assurance argument include:

— definition of the technical system context of the encompassing system;

— definition of the set of environmental conditions and operating context for which the assurance argument is valid;

— potential causes of contributing AI errors considered as part of the assurance argument.

EXAMPLE 4 Examples of assumptions that might be discharged as separate arguments can include:

— assumptions on the usage and operational profile of the AI system;

— assumptions on the reliability of inputs to the AI system;

— assumptions on the fundamental performance potential of the chosen AI technology.

An example of an assurance argument for an AI-based vehicle function structured according to a strategy that addresses possible sources of insufficiencies can be found in Annex B.

8.5.2 Categories of evidence

The following categories of evidence in the form of work products created during the AI safety life cycle can be considered for use within the assurance argument.

a) Addressing insufficiencies in the specification of the AI safety requirements:

© ISO 2024 – All rights reserved

- Evaluation of the completeness of the definition of the environmental conditions and operating context (input space).This is used to confirm completeness requirements on training and test datasets (see Clause 9).

- Evaluation of the validity of the AI safety requirements derived from the safety requirements allocated to the AI system (from external sources). This includes traceability tosafety requirements allocated to the AI system and a review of the completeness and consistency of the safety-related properties used to define the AI safety requirements (see Clause 9).

b) Addressing performance insufficiencies in the design of the AI system:

- Justification for the selection of the chosen AI methods, AI technologies and AI system architecture. This can include references to performance benchmarks and analysis indicative of the fundamental potential of the chosen technology and AI system architecture to meet the safety requirements (see Clause 10).

- Evaluation of the effectiveness of architectural and development measures. This can include an evaluation of the ability of architectural and development measures (see Clause 10) to limit the impact of contributing AI errors in the AI model.

NOTE 1 These measures can include hyperparameter optimization (development measure) as well as monitoring components (architectural measure) that detect inconsistencies in the outputs and trigger a dedicated AI error reaction to ensure AI safety. This can include components that ensure a continuous availability of the functionality through redundancy and voting, and dynamic adaptation of vehicle behaviour based on the evaluated performance of the AI system.

- Evaluation of robustness against hardware and software faults during execution. This can include an evaluation of the impact of random hardware faults and systematic design faults (including software) on AI safety (see ISO 26262-5:2018[9], ISO 26262-6:2018 [10]).

- Evaluation of the impact of differences between the development and the target execution environment.

NOTE 2 This supports the argument that AI safety is achieved (see Clause 10), under the condition that some parts of the evaluation were performed within a development environment, e.g. software-in-the-loop tests, or using synthetic data (see Clause 12).

c) Suitability of AI training and AI test datasets. This can include an evaluation of the suitability of AI training and AI test datasets to achieve and demonstrate that the safety requirements have been fulfilled (see Clause 11).

NOTE 3 This includes evidence of the independence of the AI test datasets from AI training datasets.

d) Evaluation of the fulfilment of the safety requirements. This demonstrates the extent to which the safety requirements are fulfilled and, where necessary, provides rationale (e.g. risk analysis) on the requirements that are not fulfilled, or where such fulfilment cannot be demonstrated. This can include a quantitative evaluation of functional insufficiencies in the AI system with respect to target metrics and safety-related properties used to define the requirements (see Clause 9). Approaches to collect this category of evidence can make use of real, synthetic and/or hybrid datasets (see Clause 12).

e) Evaluation of the impact of AI errors. This can include an evaluation of specific properties of the AI system that can lead to AI errors and consequently hazardous behaviour of the system. This can be based on targeted testing and analysis approaches to evaluate the presence and magnitude of known causes of AI errors such as insufficient generalization capability and insufficient robustness. This evaluation is made

© ISO 2024 – All rights reserved

based on an analysis of potential causes of insufficiencies and AI errors in the AI system (see Clause 13) and includes a definition of a set of suitable measures to address the AI errors.

f) Addressing AI errors during operation:

— Identification and analysis of previously undiscovered AI errors. This includes the continual evaluation of the behaviour of the AI system during operation (see Clause 14).

— AI errors discovered during operation are analysed to understand their criticality, and a set of mitigation measures are identified, including a repetition of relevant phases of the AI safety life cycle.

— Re-evaluation of robustness against changes in the operating conditions over time (distributional shift). This supports the argument that the AI system maintains its safety-related properties despite reasonably expected changes in its deployment environment.

EXAMPLE An analysis of the resilience of the AI system to shifts in the distribution of its inputs or the effectiveness of architectural measures to detect out of training/test distribution conditions (see Clause 14).

8.6 The role of quantitative targets and qualitative arguments

Safety requirements allocated to the AI system (from external sources) can include quantitative risk acceptance criteria and validation targets (see ISO 21448:2022, clause 6[1]).

These quantitative targets are considered during the derivation of AI safety requirements and are used to define target metrics for the safety-related properties (see Clause 9).

A direct mapping between quantitative targets (e.g. accident rates) of the safety requirements allocated to the AI system and the safety-related properties of the AI system (e.g. robustness to small changes in inputs) might not be possible.

Safety analyses (see Clause 13) that evaluate the impact and potential causes of AI errors in the AI system can provide a *qualitative* argument that the residual risk of violation of *quantitative* targets defined in safety requirements allocated to the AI system is acceptably low.

A demonstration of the correlation between causes of AI errors, the safety-related properties and the fulfilment of the AI safety requirements increases the confidence in the effectiveness of the safety analysis and thereby the associated assurance arguments and evidence.

EXAMPLE 1 The safety analysis hypothesises that an inability to generalise on inputs outside of the training distribution leads to an unacceptably high rate of AI errors under certain conditions. An out-of-distribution detection as a post-processing function is therefore proposed as an architectural measure. To argue the effectiveness of this measure, the assurance argument demonstrates both the achieved coverage of out-of-distribution inputs as well as the actual contribution of out-of-distribution inputs to the overall contributing AI error rate of the AI system. Thus, both the effectiveness and the appropriateness of the out-of-distribution detection as a safety measure are argued.

To ensure that evidence referenced by the assurance argument provides sufficient confidence in the fulfilment of safety requirements, the following assumptions can be supported with dedicated assurance arguments and additional evidence:

a) The measurement targets are an adequate proxy for measuring the achievement of the safety requirements. There is a demonstrable correlation between the collected evidence, measurement targets of safety-related properties, and risk acceptance criteria associated with the safety requirements allocated to the AI system.

b) The approach to measuring the achievement of the target values of the AI safety requirements is appropriate. This includes assurance arguments for the applicability of methods used; and how representative and indicative the datasets are that are used to collect evidence. In particular:

© ISO 2024 – All rights reserved

1) The datasets (e.g. test inputs) are representative of the input space.

2) The datasets used to collect evidence are sufficient to detect critical classes of AI errors in the AI system, e.g. by covering known edge cases and triggering conditions.

3) The datasets used to collect evidence provide a representative indication of the actual AI error rate for all inputs satisfying the system assumption, including in the presence of unknown triggering conditions. This includes an evaluation of the statistical confidence of performance evaluations and overall coverage of the input space.

EXAMPLE 2 A method for obtaining a reliable target measurement for computer vision classification tasks based on a single image might not necessarily be applicable to object detection tasks involving the processing of real-time video streams.

8.7 Evaluation of the assurance argument

Confirmation measures according to 7.3.4 as well as methods and criteria for evaluating SOTIF according to ISO 21448:2022, 12.3 [1] can be used to evaluate the achievement of AI safety on the basis of the assurance argument and associated evidence. The confirmation of the assurance argument for the AI system can be used as a precondition for the recommendation for SOTIF release at the level of the encompassing system (see ISO 21448:2022, 12.4[1]).

In case of "conditional acceptance", the conditions required for a final release of the system can be documented in the assurance argument (see ISO 21448:2022, 12.4 [1]).

EXAMPLE 1 Restricted usage within the operational environment can be required to confirm assumptions regarding the distribution of triggering conditions. Once sufficient evidence has been gathered to support these assumptions, a final release can be accepted.

Sources of potential uncertainty in the assurance argument can be used to structure the evaluation procedure and to identify potential for strengthening the argument. This includes the identification of defeaters which might contradict assertions within the argument [11]. The following types of assertions can be identified for scrutiny within the assurance argument:

— Asserted context: These assertions are associated with the contextual information and assumptions that are used to scope the claims within the argument. If these assertions cannot be demonstrated to be valid, then the conditions under which the assurance argument is valid will be restricted.

EXAMPLE 2 Changes in the operational environment in which the AI system is deployed can undermine the assumptions made on the input space of the AI system, thus undermining the validity of the assurance argument.

EXAMPLE 3 An incomplete documentation or understanding of safety requirements allocated to the AI system will undermine the validity of the statements made within the assurance argument.

— Asserted evidence: These assertions relate to the evidence used to support claims in the assurance argument being both appropriate and trustworthy. This relates to individual pieces of evidence as well as combinations of evidence that are used to support a specific claim. If these assertions related to evidence cannot be argued with sufficient confidence, then the veracity of the claim can no longer be asserted.

EXAMPLE 4 Tests based on samples taken from within a test dataset are used to provide evidence for the robustness of the AI system against rare events (edge cases). However, a sufficient number of data points representing such events is not available, resulting in test results which are insufficient to demonstrate that the robustness claim has been fulfilled within a given statistical confidence interval. Therefore, additional or alternative forms of evidence are required.

EXAMPLE 5 Synthetic test data can be used to generate a sufficiently large and diverse number of edge cases that are not commonly found in samples taken directly from the operating environment. Additional assurance arguments can

© ISO 2024 – All rights reserved

ensure the appropriateness of the testing approach to support the claim based on a validation of the fidelity of the synthetically generated data in comparison to the target environment and the inclusion of real-world samples in the test set.

EXAMPLE 6 When tools are used to produce evidence, the level of confidence in the evidence is directly linked to the confidence in the usage of the software tools. This is achieved by applying the requirements from Clause 15. Work products from Clause 15 can be used in the assertion of the validity and integrity of evidence in the assurance argument.

— Asserted inference: These assertions relate to the reasoning behind the structuring of the assurance argument itself. In particular, how top-level claims are iteratively refined into detailed sub-claims that can be directly supported by evidence.

EXAMPLE 7 The set of causes of AI errors in the AI system used to structure a risk-based assurance argument overlook critical exacerbating factors (e.g. variation in sensor positioning and calibration) resulting in an assurance argument that demonstrates a set of properties that are not sufficient to ensure all safety requirements allocated to the AI system are met.

NOTE The use of assurance claim points [6],[11] can be used to elaborate those assertions within a GSN argument that require additional confidence arguments.

8.8 Work products

8.8.1 Safety assurance argument, resulting from8.3.1and 8.3.2.

8.8.2 Confirmation measure reports, resulting from8.3.3.

9 Derivationof AI safetyrequirements

9.1 Objectives

The objectives of this clause are:

a) to specify a complete and consistent set of AI safety requirements, that are sufficient to ensure AI safety;

b) to refine AI safety requirements based on learnings from development, verification and validation;

c) to specify the limitations of an AI system over its inputspace to be escalated to itsencompassing system development process.

9.2 Prerequisites and supporting information

The following information shall be available at the initiation of this activity:

a) AI system definition (from external sources), including:

1) safety requirements allocated to the AI system;

2) input space definition;

3) functional requirements;

4) impacted stakeholders;

5) the interfaces of the AI system with the encompassing system, including if applicable, the ASIL capability of the inputs to the AI system;

6) interfaces to the environment, if applicable.

© ISO 2024 – All rights reserved

NOTE Safety requirements allocated to the AI system are allocated from the encompassing system development process. They can be motivated by different aspects, e.g. functional safety, SOTIF, or indirectly due to security (e.g. robustness against adversarial attacks and data poisoning). In this document, safety requirements are not explicitly distinguished by these aspects.

The following information can be considered during further iterations of this activity:

b) safety analysis report, from Clause 13;

c) evaluation report of functional insufficiencies detected during operation, from Clause 14;

d) measures for ensuring AI safety during operation, from Clause 14.

9.3 General requirements

9.3.1 The input spacedefinition of the AI system shall be refined to the degree suitable for initiating the AI safety lifecycle.

9.3.2 To provide a connection between each AI safety requirement and the addressed problem, the refined AI safety requirements shall either:

a) trace to the safety requirements allocated to the AI system (from external sources), assumptions or critical scenarios; or

b) address and trace to the potential influencing factors or root causes of functional insufficiencies and triggering conditions.

9.3.3 A justification shall be provided that the refined AI safety requirements are reasonable to either ensure the achievement of the safety requirements allocated to the AI system (from external sources), or prevent or mitigate the functional insufficiencies at the AI system level.

NOTE Refined AI safety requirements to address functional insufficiencies at the AI system level are identified by safety analysis as necessary to fulfil the safety requirements allocated to the AI system (from external sources).

9.3.4 To argue for the absence of unreasonable risk due to random hardware faults and classic systematic faults, the requirements of the ISO 26262 series shall be fulfilled.

NOTE 1 Combining 9.3.1 to 9.3.4, all causes defined in Figure 6–12, i.e. functional insufficiency (9.3.1to 9.3.3) and contributing systematic faults and contributing random hardware faults (9.3.4), can be addressed.

NOTE 2 Some adaptions can be necessary, in particular for safety requirements motivated by ISO 21448 activities with no ASIL rating and since the target is to achieve AI safety and not only functional safety.

9.3.5 The following cases shall be identified and reported to the encompassing system development process:

a) the AI system does not fully comply with the AI safety requirements;

b) the AI safety requirements are only fulfilled for a limited part of the input space.

9.3.6 AI safety requirements shall be identified to support the measures ensuring AI safety during operation.

EXAMPLE Different goals of field observation can be addressed with requirement 9.3.6:

© ISO 2024 – All rights reserved

a) monitoring of the uncertainty in the current situation to indicate this to the encompassing systems allowing for adjustments of the driving mission tactical control of the vehicle;

b) monitoring of performance and detection of failure to support mitigation measures internal to the AI system (also referring to Clause 10.5);

c) supporting the continuous improvement of the AI system to ensure AI safety (e.g.recording devices toidentify inconsistency among different sensor modalities and collect data for training and updating AI models);

9.4 General workflow for deriving safety requirements

Figure 9–1 explains general requirements in Clause 9.3 for deriving AI safety requirements and establishes their connections to the objectives.

— The *safety requirements allocated to the AI system (from external sources)*are part of the AI system definition provided by the encompassing system development process. These safety requirements have SOTIF and functional safety aspects and are refinedinto an initial set of AI safety requirements. SOTIF requirements are typically identified as allowed maximum error rates while being exposed to the input space.

NOTE 1 "*Safety requirements allocated to the AI system (from external sources)*" are not work products in this document. "*AI safety requirements*" refersto all requirements derived within the scope of this document.

— Refined AI safety requirements (quantitative or qualitative) will be derived either by referencing requirements from past products, or by utilising safety-related properties of AI systems that can be relevant to the application. These requirements will control the uncertainty in the development process of the AI system in order to achieve the development quality of AI models and in addressing the safety-related issues in the AI system in order to achieve SOTIF. As the work product 9.6.2*AI safety requirements*, these requirements are distributed to further development tasks in different phases of AI safety lifecycle as described in other clauses (see Clauses 9.5.1 and 9.5.2).

NOTE 2 Organizations define criteria to evaluate the uncertainty in the AI development process, i.e. rigour in training, evaluation, data collection, labelling, etc., to achieve the integrity of AI models, the safety-related errors in the AI system, and their overall impact to the encompassing system's safety. These criteria capture the organizational acceptance of the uncertainty in the AI development process and the safety-related errors in the AI system. Thus, it guides requirements elicitation, development tasks and decisions.

NOTE 3 Derived AI safety requirements include SOTIF and functional safety aspects and are allocated to AI components (See Clause 10), AI systems, encompassing systems, systems, vehicles, mobility services, etc. SOTIF requirements do not have ASIL values. Only functional safety requirements have ASIL values, including QM, and comply with the applicable parts of the ISO 26262 series.

NOTE 4 Derived AI safety requirements can be allocated to the AI system or the AI system and further to AI components and tested at the respective level. In particular, the requirements allocated to the AI components, with appropriate safety metrics and test targets, are derived from the requirements allocated to the AI system. The derivation of the requirements allocated to AI components considers the complexity of the involved AI models, their task, and the environment they operate in.

EXAMPLE 1 The AI safety requirements allocated to an AI system for automatic braking, including test targets, can be derived from the vehicle-level behaviour related to vehicle speed, distance to an obstacle, etc. If the AI system is composed of multiple AI components, the AI system decomposition needs to be accounted for in the derivation of the requirements allocated to the AI components. Examples of such decompositions are parallel AI model ensembles, parallel heads in multi-task architectures, and sequential components in multi-stage object detectors.

— The *input space definition*, e.g. ODD in the automated driving context,in the prerequisite 9.2, list item a)*AI system definition (from external sources)* is provided by the encompassing system development process.

© ISO 2024 – All rights reserved

The definition can require further refinement to be used in AI safety lifecycle, as described in Clause 9.5.3 and distributed as the work product 9.6.1*input space definition (refined)*.

— For functional safety requirements with ASILs (including QM), compliance to the ISO 26262 series can be demonstrated. For SOTIF requirements, the occurrence of particular error (sequence) patterns might be too high with respect to the criteria used to evaluate the uncertainty of the AI system during the engineering of the AI system or during operation. Refined AI safety requirements can be derived to inhibit the error (sequence) patterns produced by the AI system by understanding the triggering conditions or the safety-related errors. When further performance improvements are unlikely, performance limitations and relevant triggering conditions can be reported to the encompassing system development process. This is described in Clause9.5.4.

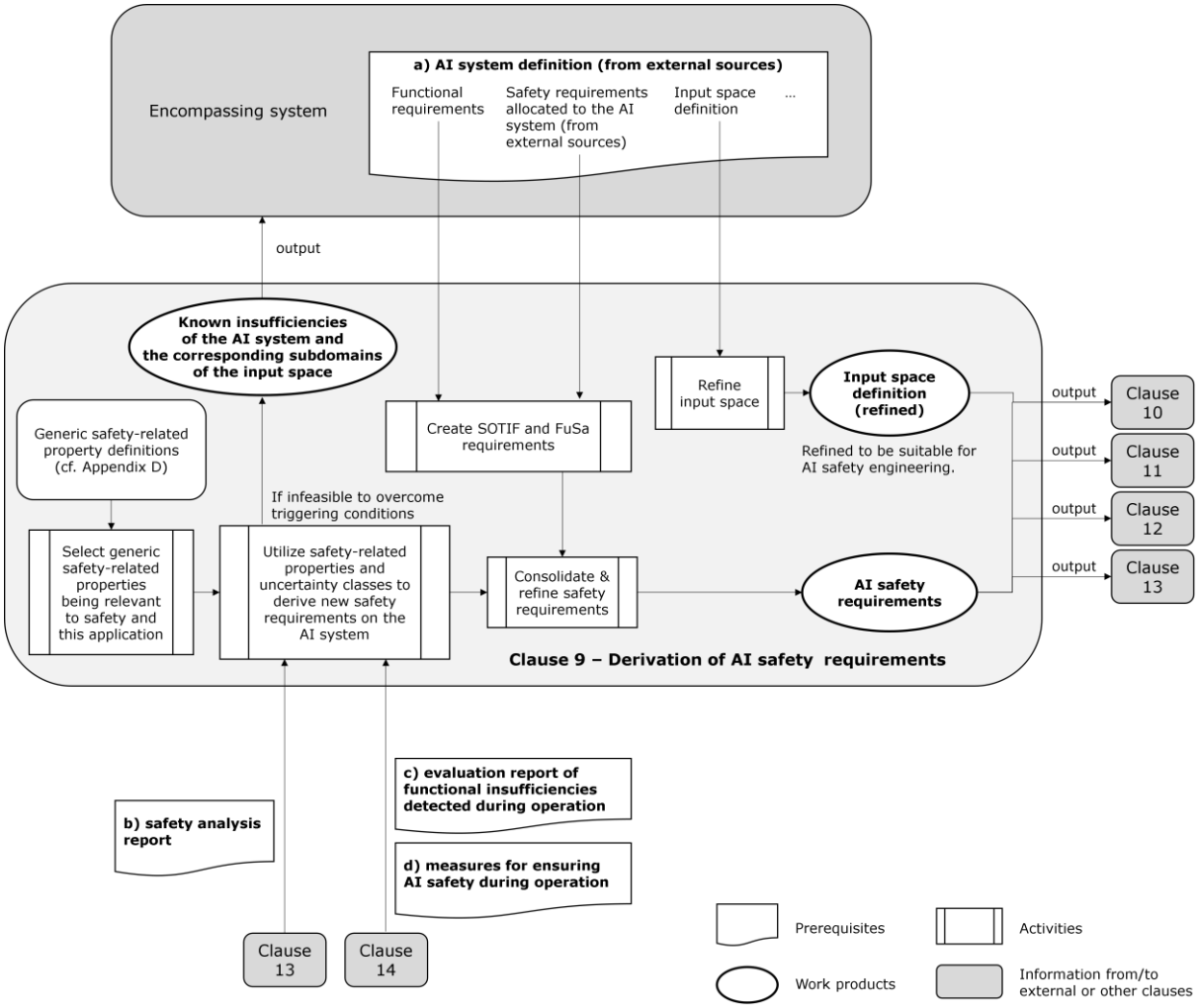
— Safety analysis (from Clause 13) or observations from the field (from Clause 14) can be used to determine the required thresholds for particular error (sequence) patterns.

— The activities described in Clause 13 either evaluate the residual risk of the AI system with respect to the AI safety requirements, or identify the safety-related errors in the AI system which can lead to the violation of the safety requirements allocated to the AI system (from external sources). The work products resulting from Clause 13 activities will be used as an input for a subsequent iteration of Clause 9 activities.

EXAMPLE 2 A lane detection function produces AI errors during night-time and heavy snow. Continued operation under these conditions case could lead to the violation of AI safety requirements. This information can be communicated to the encompassing system development, so that measures can be taken at the system or vehicle level to restrict the conditions of operation or otherwise mitigate against AI errors under these conditions.

— During the operation phase, field monitoring can be used to detect unknown triggering conditions and violations of assumptions.

© ISO 2024 – All rights reserved



NOTE For ease of understanding, the following activities are omitted in the diagram:

a) reviewing the selection ofsafety-related properties of AI systems based on safety analysis in an iterative manner;

b) refinement of input space definition based upon e.g. safety-related errors or component-level triggering conditions;

c) reporting to the encompassing systemdevelopment process forthe technical limitations of the AI system to ensure the AI safety requirements and the safety requirements on the encompassing systems and the vehicle reach consistency.

Figure 9–1 — Conceptual diagram reflecting major activities in derivation of AI safety requirements

9.5 Deriving AI safety requirements on supervised machine learning

9.5.1 The need for refinedAI safety requirements

Based on the ISO 21448 (SOTIF) activities at the vehicle level to identify and evaluate risks, potential functional insufficiencies, and potential triggering conditions, safety requirements allocated to the AI system (from external sources) are refined into AI safety requirements.

For AI systems operating within a specific input space, AI safety requirements can contain acceptance criteria similar to the concept of probability of failure on demand (PFD)[12], e.g. “Probability (occurrence of an error pattern) < α”.

© ISO 2024 – All rights reserved

EXAMPLE 1 An AI error (sequence) pattern of a DNN “Consecutive misdetection (False Negatives, FNs) of a nearby pedestrian for more than 0. 1 seconds” can be refined into “The occurrence of more than three consecutive FNs of a pedestrian @30FPS” if the DNN is triggered at a rate of 30 FPS. Such an AI error (sequence) pattern with FNs can be verifiable only if the ground truth is available.

Methods such as Systems-Theoretic Process Analysis (STPA) can be used to refine AI safety requirements where a detailed design is available (see Annex E), i.e. low-level control structure in STPA terms. However, specific types of AI models, e.g. those implemented by supervised learning, cannot be decomposed further. This clause focuses on the case that the AI model cannot be decomposed.

EXAMPLE 2 An example detailed design in an AI system can be:

— redundant DNNs for perception task;

— a majority voting on the individual results;

— an out-of-distribution (OOD) detector to abort the validity of the majority voting upon detection of an out-of-distribution input.

Based on such a detailed design, refinedAI safety requirements can be derived to address unsafe conditions, e.g. OOD detector provides an incorrect output in rainy situations.

For machine learning applications, the probability of an error in an AI system may not be computable due to the complexity of its input space, e.g. pedestrian detection for a wide variety of pedestrians in automated driving, and the inherent nature of machine learning, i.e. in-distribution and out-of-distribution error gap. The relative frequency of an error in an AI system often can be estimated only if enough samples are observed in its input space. The inaccuracy of relative frequency, e.g. PFD, is the uncertainty of an AI system that refined AI safety requirements can control in the AI development process. Requirements on a sufficiently low level of probability of AI errors are refined into quantitative requirements and qualitative requirements. Figure 9–2 illustrates the underlying approach.

EXAMPLE 3 Asafety requirement allocated to an AI system (from external sources) “Probability (occurrence of two consecutive FNsof a nearby pedestrian per hour of operation) < 10^{-6}” can be refined into an AI safety requirement “The DNN does not produce two consecutive FNs of a nearby pedestrian in 10^{8} hours of driving data with sufficient diversity demonstrated” under the condition the validity of the refinement is provided. Itis assumed that more driving hourscan be used to prove the hypothesis that the requirement holds.

NOTE 1 Relative frequency, empirical probability or experimental probability of an event is the ratio of the number of outcomes in which a specified event occurs in the total number of trials [13].

NOTE 2 For estimating very low probability using relative frequency, the denominator, i.e., the number of samples to be tested, can be large. Sample size determination for estimating the probability of occurrence of a particular error pattern can be based on estimation principles and the statistical confidence of experimental results.

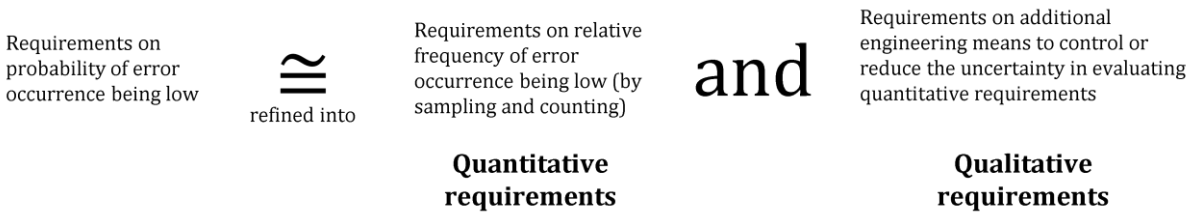


Figure 9–2 — Understanding refinement of AI safety requirements for supervised learning

© ISO 2024 – All rights reserved

9.5.2 Derivation of refined AI safety requirements to manage uncertainty

For engineering machine learning components, risks can be reduced by managing sources of uncertainty in the AI engineering process. Uncertainties exist for each phase of the AI engineering process, which arereferredto here as potential *influencing factors*. Influencing factors fall into the following categories: Observation, Label, Model, and Operation. Table 9–1describes typical approaches to managing influencing factors in the AI engineering process, which constructs an abstracted framework to reduce the uncertainty regarding limited knowledge about the true state of the world.

NOTE 1 Such limited knowledge is expected in any data-driven AI developments. Perceptual uncertainty refers to uncertainty associated with the performance of perceptual components[14].

Table 9–1 — Managing influencing factors in the AI engineering process

Influencing factor class	Uncertainty management approach
Observation	(1) Enumerate patterns influencing AI outputs both in deductive (top-down) and in inductive (bottom-up) ways (2) Define data coverage policy for comprehensive AI performance and safety (harm avoidance) trade-off
Label	Reduce variation of labelling policy and enhance/ensure labelling work quality
Model	Use appropriate model selection and de facto standard approaches for tuning and evaluation
Operation	Detect deviation in model performance between development and deployment

Addressing the above influencing factors, Table 9–2describes a non-exhaustive set of example generic AI safety requirements that may be considered for specific applications.

Table 9–2 — Example generic AI safety requirements to manage influencing factors

Influencing factor class	Example generic AI safety requirements
Observation	— Derive scenes relevant to attributes of inference targets, environmental conditions, and system configurations (relevance) — Include scenes which would potentially lead to errors (e.g., false negative, false positive, substantially incorrect regression) — Consider the effects of system configuration change during operation (e.g., sensor mounting position shift, sensor hardware degradation caused by ageing, and suitable sensor parameter change due to it) — Include specific scenes which empirically have led to errors during development of the current product and development and operation of legacy products (safety) — Specify data quantity for each scene or combination of scenes (diversity) — Explain the rationale for the scenes for which one cannot define target data quantity
Label	— Specify inference targets — Describe the necessary quality of labels, e.g. false positive rate, bounding box size precision, etc.

© ISO 2024 – All rights reserved

Influencing factor class	Example generic AI safety requirements

	— Specify labelling procedure
	  — Define the appropriate type of label e.g. class, numerical etc.
	  — Publish and circulate a labelling policy guideline, e.g. treatment of occluded objects, number of annotators annotating the same data etc., to all relevant stakeholders
	  — Provide clear criteria and lines of accountability about the labelling of data involving protected characteristics or special category data (or both)
	— Specify label evaluation policy, i.e., evaluation timing, evaluation procedure, evaluation measures, and acceptance criteria
	  — Make sure appropriate processes are in place if crowdsourced labellers are used
	  — Assess the risk of incorrect labelling through sample checks of submitted labels
	— Specify label incident reporting and management process to handle labelling errors and ambiguity in labelling policy

Model	— Specify metrics and acceptance criteria for model performance — Specify model selection policy and ensure an appropriate model is selected according to the policy — Specify hyperparameter search policy and conditions, e.g., bounds, and document the determined hyperparameter values — Perform safety analysis of the AI model — Introduce the technical capability of checking the plausibility of the output — Introduce the technical capability of de-noising/quantizing the input — Introduce the technical capability of robust training techniques (e.g., training against noise, quantized neural networks) — Introduce the technical capability of providing interpretation (explainability) over the decision being made — For DNN, pre-hoc global explainability (i.e., extracting the decision-making mechanisms into human-comprehensible rules) can be difficult. Post-hoc local explainability (i.e., the reason why a particular decision is made for a specific input) is likely, via techniques such as local linearization or heatmaps [15] — Introduce the technical capability to measure epistemic uncertainty for the AI model outputs

© ISO 2024 – All rights reserved

Influencing factor class	Example generic AI safety requirements
Operation	— Introduce the technical capability of identifying situations matching a known triggering condition or measuring model uncertainty — Introduce the technical capability of identifying concept/domain drift — Introduce the technical capability of identifying implausible or otherwise non-trustworthy AI system behaviour

While AI safety requirements derived using influencing factors are largely **qualitative**, how safety-related properties of AI systems can be used to assist in derivingrefined**quantitative**AI safety requirements is considered in the following.

In this document, we establish a differentiation between safety-related properties and AI safety requirements.

— Safety-related properties are inherent characteristics of engineering AI systems which may either lead to the insufficiency of the generalization or merely make it difficult to argue the safety of the AI system. Safety-related properties are a subset of generic AI properties; they are application-independent and may include aspects such as robustness or domain shift, incomplete specification as suggested by ISO/IEC 22989 and related guidance on safety in ISO/IEC DTR 5469. See Annex Dfor a list of AI properties that may be safety-related.

— AI safety requirements are application-dependent and can be mapped to one or more safety-related properties.

NOTE 2 The term safety-related property used in this document is stated differently in various project contexts with similar meanings. Within the French national project DEEL, it is referred to as high-level properties[16], while in the German national project KI-Absicherung (EN: AI-assurance), it is referred as safety concerns[17].

The list of safety-related properties of AI systems from Annex D can be used to refine AI safety requirements. The safety-related properties of AI systems considered relevant for the AI system under consideration are identified and instantiatedinto refined AI safety requirements. These requirements can be associated with the safety-related properties of AI systems to support the selection of AI technologies, and architectural and development measures, as detailed in Clause 10.

Deductive safety analysis approaches such as FTA might not directly lead to the identification of safety-related properties and associated AI safety requirements. This is because a causal analysis of AI errors might have limited effectiveness due to the complexity of the model and interaction of numerous, potentially unknown causes. Instead, inductive approaches such as hypothesis testing are used to derive the safety requirements. First, an initial set of AI safety requirements is identified, and then AI safety requirements are updated and validated through safety analysis.

NOTE 3 These safety-related properties of AI systems are currently described conceptually rather than using formal definitions in mathematics. The list of safety-related properties on AI systems, e.g., Annex D, is not exhaustive. The list is based on discussions in AI safety research, etc.

Refined AI safety requirements can be either qualitative or quantitative, where for quantitative AI safety requirements, the design of the acceptance thresholds is application dependent. The validity of these thresholds requires justificationwith respect to the evaluation criteria and the overall residual risk acceptable for the AI system. For the example of robustness, the validation means “Can setting such a threshold positively increase the robustness?”, which is an activity to be evaluated via experiments within Clause 12.5.7.

EXAMPLE Appropriate safety-related properties are hypothesized at the early stage of development, e.g., based on past product experiences and identified by safety analysis in an iterative manner to ensure their contribution to the system‘s safety. Safety analysis can include the impact of changing safety-related properties, specific noise robustness for the equipped hardware characteristics, etc., on the system’s safety. For example, a safety-related property of AI systems AI robustness is identified by such safety analysis and further refined to different kinds of robustness, then the following

© ISO 2024 – All rights reserved

quantitative AI safety requirement can be derived along with justification for the validity of these thresholds: “For all clear images in the input space, if noise perturbations characterized by L_1 norm < 0.001 are added on the image, the AI system should at most introduce 0.01% of new errors”. This requirement, derived from “AI robustness,” will lead to additional engineering efforts in Clause 10, such as using robust training techniques.

Using safety-related properties of AI systems to derive refined AI requirements is an inductive approach, and exhaustiveness is not ensured. Therefore, it can be only a complementary part of the verification and validation strategy elaborated in ISO 21448 (SOTIF) Clause 9.

9.5.3 Refinement of the input space definition for AI safety lifecycle

The input space defines what workspace, what conditions, and around what dynamic elements a system will operate with the purpose to ensure its safe design and safe operation. If a system depends on AI systems using data-driven methods to build machine learning models, the input space leads to the definition of the dataset requirements, which form the basis of the data used throughout the lifecycle of the system.

EXAMPLE For autonomous driving, the workspace can be the road or area where the system operates. The conditions can be the weather, visibility conditions, illumination, and connectivity. The dynamic elements can be surrounding traffic and moving debris.

The refinement of the input space is an iterative process. The initial definition provided as a prerequisite may not be suitable to be used for ensuring the intended functionality of the AI system. This is also reflected in the ISO 21448 document, emphasizing that environmental factors are essential issues, while systems and their elements have different concerns depending on the hierarchical layers.

The refinement of the input space definition aims at complying with the capabilities of the available systems (e.g., sensors). Its refinement may also consider newly discovered or known triggering conditions that may lead to performance insufficiencies. Regarding where and what kind of refinement of the definition of the input space is needed, the information can be derived from the results of the safety analyses as described in Clause 13.

The refined input spacedefinition forms the basis for the verification and testing of the target system performance. That is, the refined input space bounds the performance expectations of the AI system, reducing the uncertainty associated to its operation and consequently ensuring the bounds of its safe operation.

9.5.4 Restricting the occurrence of AI output insufficiencies

Overall, during the engineering of an AI system, a particular scenario (or a set of scenarios) where the error rate of the AI system is too high may be observed, violating the initially derived AI safety requirements. The predefined list of safety-related properties of AI systems can be used again as a checklist to examine the potential root causes and subsequently, derive a new set of AI safety requirements with associated metrics. In the iterations beyond the initial iteration, the safety analysis described in Clause 13 provides additional input to this activity. Based on observations from the field or from V&V activities, the safety analyses identify the safety-related properties which maximally correlate with the safety-related issue. The insights into the correlation (or if applicable causality) between output insufficiencies and safety-related issue can be used as a basis for the refinement of the AI safety requirements after the initial iteration.

NOTE 1 Currently these safety-related properties are described conceptually rather than using formal definitions. The consequence is that the derivation of AI safety requirements is not viewed as a "(logical) causal derivation" but rather a hypothesis to be validated.

NOTE 2 The vast majority of ML-type statistical models estimate the likelihood of events, which is inferred from the correlations in the data. In such cases clear causal relationships between the inputs and the outputs of these models (and ultimately the actions taken by the system) cannot be established. Thus, the classic causal fault analysis tools are not necessarily applicable in this context.

© ISO 2024 – All rights reserved

In addition to the analysis results from Clause 13, which are driven by V&V issues or issues observed in the field, analytic methods can also support the creation and refinement of AI safetyrequirements, which may include plausibility checks on the inputs to the AI system and bounding on the outputs from the AI system.

As illustrated in Figure 9–3, an issue associated with performance insufficiencies may be identified from a too-high error rate during operation. By conducting appropriatelydesigned experiments (e.g., counterfactual analysis or hypothesis testing), it may be discovered that the issue is strongly correlated with one or more intrinsic AI properties that are safety-related (e.g., robustness), with evidences of correlation empirically manifested in some measurable metrics. The activity can be continued to further refine the influencing factors as described in Table 9–3, thereby deriving reasonable AI safety requirements with the aim to prevent or mitigate the performance insufficiencies. The effectiveness of the refinedAI safety requirements as counter-measures should still be validated (see Clause 12.5.7).

NOTE 3 The use of correlation-driven analysis techniques is driven by the practical need as demonstrated in the field of supervised machine learning. It does not inhibit the establishment of an assuranceargument for AI, provided that additional supporting evidence such as experiments can be provided. This document also does not inhibit logical causal analysis for AI as used in establishing the safety case, provided that causality can be rigorously demonstrated.

NOTE 4 Regarding systematic errors, technical safety requirements allocated to the AI system are achieved by safety mechanisms as the countermeasures for systematic errors.

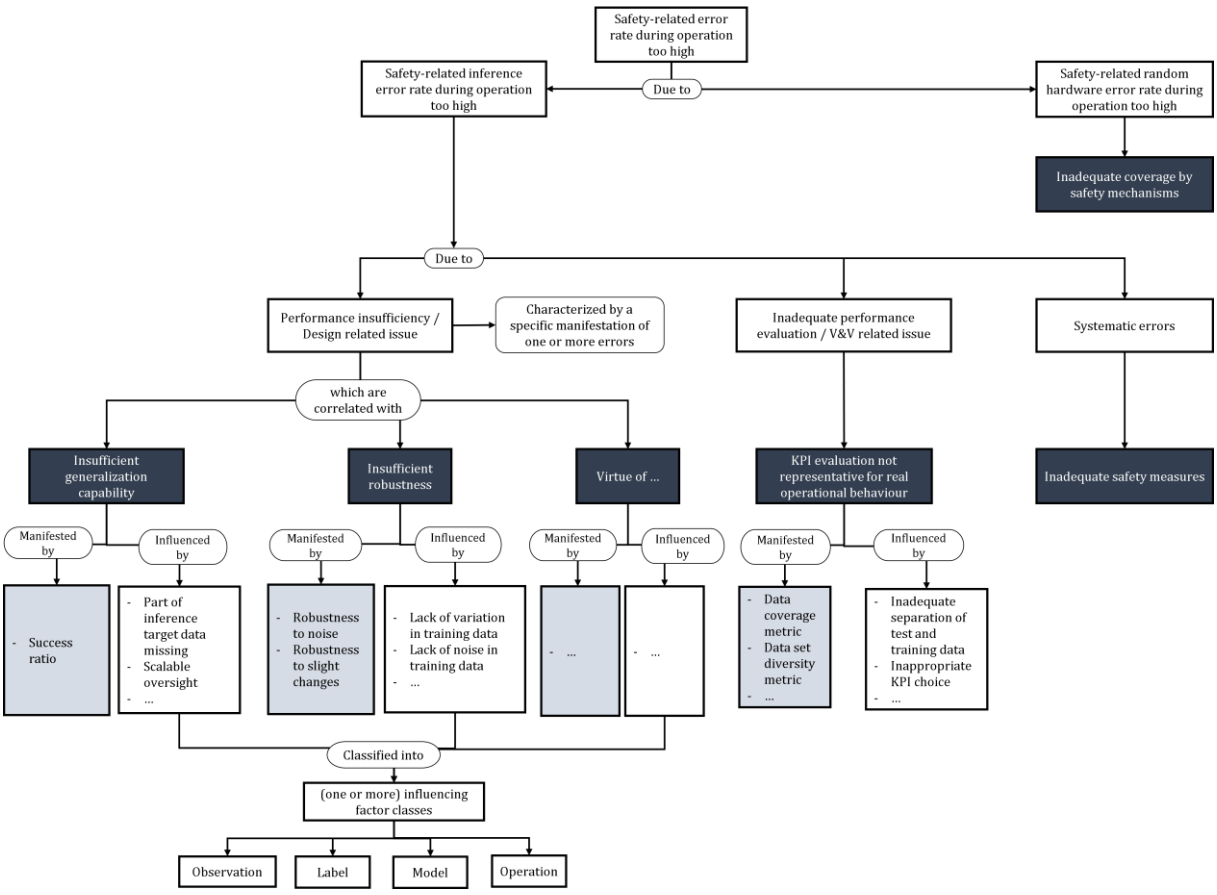


Figure 9–3 — Conceptual diagram illustrating safety-related properties

Table 9–3 — Influencing factor classes

Influencing factor class	Description	Examples
Observation	Influencing factors related to data representing the input space	— Diversity of training and test data

© ISO 2024 – All rights reserved

Influencing factor class	Description	Examples
		— Coverage of the inference target data domain — Distribution according to features of the inference target domain
Label	Influencing factors related to the data labels	— Incorrect labels — Inconsistent labels
Model	Influencing factors related to the ML model itself	— Choice of ML model — Choice of hyperparameters — Training procedure — Limited computing power/timing/memory, or precision change/constraint during deploying AI systems into target environment, e.g., hardware NOTE The impact of differences due to deployment constraints between the AI model developed in the off-board environment and its on-board implementation cannot be analytically determined due to the complexity of AI models and is left as uncertainty in the AI context.
Operation	Influencing factors related to changes within the input space during operation	— For a ML system classifying traffic signs: Introduction of new traffic signs during its operational phase

Table 9–4 illustrates an example of connecting insufficiencies, safety-related properties, and concrete AI safety requirements.

Table 9–4 — Examples for utilizing safety-related properties to derive new safety requirements

Insufficiencies observed from the field	Associated safety-related properties	Concrete AI safety requirement derived from insufficiencies	Metrics as KPIs	Original error pattern being considered
Under slight input variations, the object being detected is missing (false negative)	Robustness (inherent in model training, decision boundary)	For any training data, if a Gaussian noise bounded by L-infinity norm < 0.01 is added, the predicted output class should remain	For each image in the training/test data, evaluate the robustness by creating 1000 variations of each image with additional Gaussian noise, and derive	The safety requirements allocated to the AI system (from external

© ISO 2024 – All rights reserved

Insufficiencies observed from the field	Associated safety-related properties	Concrete AI safety requirement derived from insufficiencies	Metrics as KPIs	Original error pattern being considered
		the same with a high probability (e.g., 99.5%)	the number of incorrect predictions. KPI returns "violation / false" if there exists an image where the number of incorrect predictions subject to noise is larger than 5.	sources)related to false negatives
When HW faults occur where a small area of pixels has turned black, the prediction can be constantly wrong (false negative)	Robustness (against random HW faults)	For any training data, if 1% of the pixels is arbitrary changed to be completely black, the predicted output class should remain the same with a high probability (e.g., 99.5%)	For each image in the training / test data, evaluate the robustness by randomly setting 1% of the pixels to black, repeat 1000 times, and derive the number of incorrect predictions. KPI returns "violation / false" if there exists an image where the number of incorrect prediction subject to noise is larger than 5.	The safety requirements allocated to the AI system (from external sources)related to false negatives
NOTE L-infinity norm quantifies noise by taking the maximum absolute value of the noise on every input dimension. For example, if there are three inputs to the DNN and the noise on each input is 0.1, -0.2, and 0.15 respectively, then L-infinity norm = max {|0.1|, |-0.2|, |0.15|} = 0.2. The use of L-infinity norm is only an option; there exists other norms (e.g., L1 or L2) for quantifying the noise.				

9.5.5 Metrics, measurements and threshold design

Quantitative evidence requires the definition of a set of metrics and target thresholds. The metrics are used to define, characterise and theoretically analyse specific properties of AI systems. The selection and design of these metrics and associated measurements and target thresholds involves the following considerations:

— Metrics and measurement methods for specific properties of AI technologies referenced in the safety requirements allocated to the AI system (from external sources).

— Measurements and evaluation of the AI systems with respect to these properties.

— Analysis of how the AI methods used to develop the AI system impacts themetrics/measurement methods chosen for the AI safety requirements.

— Definition and creation of suitable datasets used to measure each property (see Clause 11). The datasets are designed to provide a statistically relevant analysis of the property and be representative of the input space.

Selection of suitable target thresholds for each metric requires justification. The justification can include:

— past product experiences;

— commonly agreed industry consensus;

— system analysis;

— expert judgements;

© ISO 2024 – All rights reserved

— experiments.

The decision may also involve multiple factors where trade-offs (e.g., bias-variance trade-offand robustness-accuracy trade-off phenomena in machine learning) are taken into consideration.

EXAMPLE As an example, when one considers robustness against noise demonstrated by the minimum perturbation to change to incorrect prediction, the fundamental question is why one uses ε_1 (e.g., 0.5) rather than ε_2 (e.g., 0.25) as the acceptance criteria. There can be multiple ways of justifying the threshold by providing convincing arguments, e.g., the robustness should be larger than ε_1, as

— ε_1 is the maximum possible noise communicated from the complete input pipeline, or

— ε_1 is derived from experiments where experts agree that noise exceeding ε_1 is also hard for human to make reasonable prediction, or

— ε_1 is the value being used in similarstandards or products.

9.5.6 Considerations for deriving safety requirements

While previous sections offer the general principles in deriving refined AI safety requirements via the assistance of safety-related properties of AI systems, the following is a summary of some practical example considerations for the derivation of AI safety requirements.

NOTE This list of considerations is not exhaustive. Nevertheless, together, they address complementary topics that can be relevant to achieve safety.

a) Given the lack of guarantees on the generalization ability of ML models, it is important to specify the input space defining the workspace, conditions, and dynamic elements, thereby limiting the performance requirements and also the necessary training data to that space.

b) Results from statistical learning theories are commonly based on idealistic assumptions. These theories (e.g., Vapnik–Chervonenkis theory[18]) can nevertheless be used to derive a lower bound on the number of samples needed to ensure a tight generalization error for a given type of ML model (e.g., support vector machine). The derived number of samples for model training commonly reflects the best-case scenario where the data used in training (in-sample) has the same distribution as the data in operation (out-of-sample).

c) The possibility of relevant foreseeable adversarial attacks, i.e., foreseeable attacks that are judged to be realistic, and their impact on the overall AI system can be considered. However, cybersecurity has not been considered in this document.

d) The side effects of region-specific privacy considerations (e.g., GDPR in EU) can be considered, as they can indirectly influence the quality of the collected data and impact the resulting model performance.

e) A practical purpose of improving AI explainability is to ease the engineering of AI systems. For validation and performance improvement purposes, interfaces to AI components can be specific to improve the understanding of the AI component response. Such interfaces can allow for introspective approaches for validation and performance improvement, particularly isolating errors of the black-box machine learning based AI components.

f) Toreduce gaps or non-conformities to a particular AI safety requirement(e.g., by improving the related performance),different performance aspects, which are inherent to the selected AI methods, can become entangled. If overall efforts cannot be increased to meet either of the conflicting safety requirements, a trade-off (e.g., between robustness and accuracy) is found. As shown in Figure 9–1, such limitationscan be forwarded to the encompassing system development process so that safety requirements on the AI

© ISO 2024 – All rights reserved

system and the encompassing system design are updated, and the entire safety requirements are finally satisfied.

g) Results from the Neyman-Pearson approach (Hypothesis testing) can be used to minimize the missed detection (number of false negatives) under a fixed number of false alarms (number of false positives) by considering inputs like sample size and signal-to-noise ratio (SNR).[19]

9.6 Work products

9.6.1 Input space definition (refined), resulting from 9.3.1.

9.6.2 AI safety requirements, resulting from9.3.2, 9.3.3, 9.3.4, and9.3.6.

9.6.3 Known insufficiencies of the AI system and the corresponding subdomains of the input space, resulting from 9.3.5.

10 Selection of AI technologies, architectural and development measures

10.1 Objectives

The objectives of this clause are:

a) to select and justify appropriate AI technologies for use in the AI system;

b) to identify appropriate architectural anddevelopment measures tofulfil the safety requirements prior to deployment;

c) to identify appropriate architectural measures to mitigate residual functional insufficiencies of the AI system revealed after deployment;

d) to identify measures for ensuring the safety requirements of the AI system are fulfilled within its target execution environment.

10.2 Prerequisites

The following information shall be available:

a) safety requirements on the AI system, from Clause 9:

b) training and validation datasets, fromClause 11;

c) AI component or AI system architecture, if already existing;

d) AI component or AI system development process, if already existing.

10.3 General requirements

10.3.1 A justification shall be provided that the selected AI technologies and AI methods are capable of fulfilling the AI safety requirements.

NOTE AI technology, its application to road vehicle functionality, as well as methods for assuring the AI safety requirements, are rapidly evolving. However, the most advanced technologies might not be the most suitable for safety-related applications due to the lack of an appropriate set of methods for assuring the safety of such technologies (see Technology class III of ISO/IEC TR 5469 - [4]).

© ISO 2024 – All rights reserved

EXAMPLE After analysing the benefits and limitations of alternate technologies, an argument is made to justify why DNNs are selected and used in combination with a set of redundancy and monitoring measures for a potentially safety-related functionality despite the challenges of demonstrating safety requirements for such approaches.

10.3.2 AI safety requirements shall be allocated to AI components.

NOTE 1 In some exceptions, AI components might not have allocated AI safety requirements, e.g. diagnostics components.

NOTE 2 The AI safety requirements allocated to the AI component depend on the functionality of the AI component and on the AI safety requirements of the system.

10.3.3 Sufficient measures, such as architectural, development, or a combination of them, shall be defined to ensure the AI safety requirements are fulfilled by the AI components.

NOTE The architectural and development measures contribute to preventing, by design, AIerrors of the AI components.

10.3.4 Sufficient measures, such as architectural, development or a combination of them, shall be defined to reduce the risk resulting from contributing AI errors of the AI components.

EXAMPLE For out-of-distribution error detection, implementation of reject classes implies development measures as well as architectural measures for ML systems.

10.3.5 The effectiveness of the chosen combination of architectural and development measures resulting from 10.3.3 and 10.3.4 shall be supported by an argument.

10.3.6 Safety analysis of the AI system outputs and, where reasonably practicable, of its architectural elements shall be performed to determine whether the safety requirements allocated to the AI system can be met.

NOTE For DNNs, safety analysis of the architectural entities of the AI model might not be reasonably practicable.

EXAMPLE Such safety analysis can include the analysis of the computational graph of AI components to identify if the intermediate(latent space) or final outputs fulfils the relevant AI safety requirement allocated to the AI components.

10.3.11 The differences between the development environment and the target execution environment shall be identified and evaluated regarding their potential impact on the safety requirements and, if necessary, appropriate AI architectural and development measures shall be defined.

10.3.12 AI components that are AI models or contain AI models shall be trained using the training dataset and evaluated using the validation dataset.

10.4 Architecture and development process design or refinement

The architecture of an AI component or AI system isupdated, if available, or designed, to fulfil the safety requirements provided as input to this clause. Similarly, an AI system or AI component development process is tailored or designed.

The AI safety requirements are satisfied by two types of measures, architectural and development; there are two categories of AI safety requirements, some derived from safety requirements allocated to the AI system (from external sources) and some derived from safety related properties of AI system. For the latter ones, Table 10–1 provides some examples of measures that can help fulfil them.

It is possible that, during the design process, there is no architecture and/or development process that can fulfil all AI safety requirements. In such a case, the challenges will be discussed with the requirement

© ISO 2024 – All rights reserved

stakeholders and the AI safety requirements will be updated (see Clause 9 for guidance on updating the AI safety requirements).

Similarly, there can be more than one architecture and/or development process that fulfils all AI safety requirements during the design process. In such a case, the benefits and the cost of each option will be discussed with the system integrator to choose the most appropriate option for the AI component or AI system application.

Once a candidate architecture and development process are identified, the AI component or AI system is trained using the training dataset and its potential to achieve its allocated safety requirements is evaluated using the validation dataset.

AI model training is a critical step in the development process where the model learns from data in an iterative manner by using a preferred optimization algorithm. In "supervised learning", for example, the learning process involves learning the weights and biases that minimize the error between the model's prediction and the ground truth. The training process relies on several tunable parameters known as hyperparameters (e.g. learning rate, regularization strength, etc.). Hyperparameter tuning helps optimize the model's performance. The training process can also involve one or more of the steps such as feature engineering, regularization, dropout, error analysis etc.

10.5 Examples of architectural and development measures for AI systems

Table 10–1 provides guidance on which measures can support the achievement of the AI-property-specific KPIs and targets associated with AI safety requirements and safety-related properties of AI systems.

NOTE 1 Table D–1 in Annex D provides a definition of the safety-related properties of AI systems listed in Table 10–1.

NOTE 2 Annex G provides a short description of each architectural and development measure listed in Table 10–1.

NOTE 3 The applicability of a safety-related property of an AI system depends on the use case, encompassing systems AI models, etc. For example, while a self-driving vehicle's actions, such as acceleration and steering, could be controllable, the property of AI controllability might not apply to the outputs of a DNN model for object detection in the perception pipeline.

Acknowledging that there is an overlap between safety-related properties of AI systems, rationales used for the allocation of measures to the appropriate AI property(ies) can include:

a) *AI Resilience* supports *AI Robustness*. The following rationales were considered in the selection of recommended measures to distinguish *AI Robustness* and *AI Resilience*:

— Measures that can guarantee that the system maintains its nominal performance under bounded input perturbations are classified under *AI Robustness;*

— Measures that mitigate a failure for which the system impact is unclear are classified under *AI Resilience;*

— Some measures fall under both categories, *AI Robustness* and *AI Resilience.*

b) *AI Controllability* can support *AI Resilience*, e.g. a supervisory system detects an error and switches between redundant systems. The same supervisory system could also detect an error and trigger a risk mitigation measure that stops the AI system operation. The following rationales were considered in the selection of recommended measures to distinguish *AI Controllability* and *AI Resilience*:

— the aim of *AI Resilience* is to keep the system running;

— the aim of *AI Controllability* is to keep the system safe.

© ISO 2024 – All rights reserved

c) *AI Alignment* and *AI Predictability* share the same objective, confidence in the correctness of the AI system's prediction, but achieve it with different means:

— Measures focusing on demonstrating that the AI system's behaviour is aligned with the user's expectations and values are classified under *AI Alignment;*

— Measures providing supporting evidence that the system behaves as expected, i.e. the system's behaviour can be reasonably predicted based on its inputs, are classified under *AI Predictability.*

Table 10–1 — Example of measures fostering "AI Robustness"

Type of Measure	Measure	Remarks
Architectural Measure	Architectural measures that foster AI robustness against Out-of-Distribution (OOD) inputs (G.3.1.1)	Architectural artefacts such as reject classes can be required to detect OOD inputs.
	Diverse redundant models (G.1.1.1) Model ensembles (G.1.1.2) N-version diverse programming (G.1.1.3) Selection techniques for architectural redundancy (voting and switching) (G.1.1.5)	Architectural redundancy combined with a voting system provides confidence in the AI system generating a correct output despite some ML elements providing wrong predictions.
Development Measure	Fault-aware training (G.4.2)	Training the AI system to recognize faults helps to handle those faults whilst maintaining a nominal or a degraded mode.
	Adversarial training (G.3.1.2, G.4.2)	Adversarial training reduces the AI system's sensitivity to external malevolent perturbations and increases its reliability.
	Transfer learning (G.4.3)	Transfer learning relies on the robustness a given AI model has shown within its source input domain to leverage this robustness within the target input domain.
	Augmentation of data (G.4.9)	Using data augmentation to increase the diversity of data the model is exposed to helps it generalize better.

Table 10–2 — Example of measures fostering "AI Generalization capability"

Type of Measure	Measure	Remarks
Architectural and Development Measure	Transfer learning (G.4.3)	Transfer learning can leverage the generalization capability of the foundation model to the target application.
Development Measure	Regularization (G.4.2)	Regularization methods help the model adapt better to small shifts in the input domain and reject OOD inputs.
	Hyperparameter tuning (G.4.1)	Hyperparameter tuning can help reduce the underfitting or

© ISO 2024 – All rights reserved

Type of Measure	Measure	Remarks
		overfitting, thereby improving generalization.

Table 10–3 — Example of measures fostering "AI Reliability"

Type of Measure	Measure	Remarks
Architectural and Development Measure	All measures supporting AI Robustness, AI Resilience and AI Generalization support Reliability	None

Table 10–4 — Example of measures fostering "AI Resilience"

Type of Measure	Measure	Remarks
Architectural and Development Measure	OOD data and its mitigation (G.3.1) Distributional shifts and its mitigation (G.3.2)	Mechanisms detecting OOD samples and distributional shifts trigger the need for update and containing actions to ensure the system's safety until the OOD or distributional shift is addressed.
	Qualitative and Quantitative Analysis of AI architectures (Clause G.2)	Safety analyses on the architecture help identify the need for redundant systems that maintain the AI system to an initial or degraded performance in case of AI error.
	Usage of AI-model based and conventional software (G.1.1.6)	Non-AI components can be used to detect errors and switch to redundant or fallback systems. Redundant systems provide a means to continue the AI system operation with the initial performance.
Development Measure	Targeted and controlled model update (G.5.2.2)	Partial and targeted updates can allow simplified degradation of safety performance testing and speed up an issue resolution.

Table 10–5 — Example of measures fostering "AI Controllability"

Type of Measure	Measure	Remarks
Architectural Measure	Usage of AI components and non-AI components (G.1.1.6)	Non-AI components are used to detect errors and switch to redundant or fallback systems. Fallback systems guarantee the AI system's controllability in all situations.
	Supervisory, limiting logic and non-AI backup system (G.1.1.4)	The implementation of safety monitors provides a means to detect errors and take control over one element of the AI system to ensure the AI system's safety.
Architectural and Development Measure	Qualitative and Quantitative Analysis of AI architectures (Clause G.2)	Safety analyses on the architecture help identify the need for monitoring system that can that can detect and control AI errors.

© ISO 2024 – All rights reserved

2112 **Table 10–6 — Example of measures fostering "AI Explainability"**

Type of Measure	Measure	Remarks
Development Measure	Attention or Saliency Maps (G.4.7)	Attention/Saliency maps provide information that helps understand the characteristics of the input that strongly influence the prediction of the ML system.
	Structural Coverage of AI component (G.4.9.1)	This is a white (or open) box method; it can build confidence by identifying which features of inputs are important for the decision / prediction of AI models.
	Identification of SW units (G.2.1)	Breaking down the architecture into SW units aims to understand the function and performance of each unit, fostering some explainability of the AI component output outcome.

2113 **Table 10–7 — Example of measures fostering "AI Predictability"**

Type of Measure	Measure	Remarks
Architectural Measure	Model ensembles (G.1.1.2)Techniques for selection of architectural redundancy (G.1.1.5)	Ensembles typically increases the accuracy of the prediction, therefore fostering predictability.
Architectural and Development Measure	Criteria for Retraining (G.5.2.1)	Monitoring criteria for retraining, e.g. distributional shift, helps identify a decrease in the performance of the AI model to predict the correct output. Consequently, this monitoring also helps to detect a reduction in robustness, resilience and generalisation capability.
Development Measure	Confidence Calibration and Uncertainty Quantification of AI models (G.4.4)	Calibrating the AI model's uncertainty fosters confidence in the correctness of the model's predictions.
	Structural Coverage of AI component (G.4.9.1)	Structural coverage provides confidence in the comprehensiveness of the testing strategy.
	Monitoring multiple scores (G.4.6)	Monitoring model performance metrics such as precision, recall, F1-score during training provides insight into the model's ability to predict correct outputs consistently.

2114 **Table 10–8 — Example of measures fostering "AI Alignment"**

Type of Measure	Measure	Remarks
Development Measure	Alignment of intention (Clause G.6)	None

2115 **Table 10–9 — Example of measures fostering "Bias and Fairness"**

Type of Measure	Measure	Remarks

© ISO 2024 – All rights reserved

Development Measure	Data coverage techniques for test data augmentation (G.4.9.2)	This method can help identify and reduce bias within the datasets by measuring the data distribution across equivalence classes.

10.6 Work products

10.6.1 AI component or AI system architecture (refined), resulting from 10.3.1 to10.3.11.

10.6.2 AI component or AI system development process (refined), resulting from 10.3.1 to 10.3.11.

10.6.3 Implemented AI component, resulting from 10.3.12.

11 Data-related considerations

11.1 Objectives

The objectives of this clause are:

a) to define the dataset lifecycle of activities related to the gathering, creation, analysis, verification and validation, management, and maintenance of the datasets used in the development of the AI system;

b) to identify the dataset insufficiencies that may impact the safety of the AI system;

c) to identify the data-related safety properties that have a bearing on the safety of the AI system and that support dataset safety analysis;

d) to define the countermeasures to prevent or mitigate dataset insufficiencies using dataset safety analysis methods at different steps in the dataset lifecycle;

e) to define the data-related work products that support providing evidence of the safety of the AI system.

This clause applies to AI systems whose development and/or testing relies on data, including AI systems based on supervised, semi-supervised, and unsupervised learning techniques.

11.2 Prerequisites and supporting information

The following information shall be used at the initiation of this phase of activities:

a) AI system definition, including:

   1) AI safety requirements, from Clause 9;

   2) input space definition(refined), from Clause 9;

b) field data and functional insufficiencies detected during operation, from Clause 14;

c) safety analysis report, fromClause 13.

11.3 General requirements

11.3.1 A dataset lifecycle shall be defined for the datasets used in the development of the AI system.

© ISO 2024 – All rights reserved

11.3.2 The dataset lifecycle shall be defined such that it supports iterative development of the dataset taking into account changes in the AI safety requirements and any insufficiencies observed during the AI system deployment phase.

11.3.3 The dataset lifecycle shall include activities that relate to the gathering, creation, safety analysis, verification, validation, management, and maintenance of the datasets used to develop the AI system.

NOTE An example dataset lifecycle covers requirements development, design, implementation, verification, validation, safety analysis, and maintenance of the dataset. The dataset verification activity can ensure traceability from the dataset requirements to the dataset design and implementation. The dataset validation can involve integration of the AI system and be performed as part of the AI system verification.

11.3.4 Data-related safety properties of the dataset shall be identified and be used as inputs at different phases of the dataset lifecycle.

11.3.5 The dataset lifecycle activities shall include safety analyses to identify potential dataset insufficiencies, their root causes, and their potential to cause a violation of AI safety requirements.

11.3.6 Dataset requirements of the dataset shall

— address the dataset insufficiencies that can lead to violation of the AI safety requirements;

— specify countermeasures to prevent the dataset insufficiencies, to mitigate them, or both.

11.3.7 Traceability shall be ensured between the dataset requirements and the AI safety requirements.

11.4 Dataset life cycle

11.4.1 Datasets and the AI safety lifecycle

Datasets play a crucial role in AI system development and testing. Machine learning, in particular, typically involves an off-line training process whose purpose is to determine values for the parameters of an AI model, and three different types of datasets enable this training and its usage afterwards: AI training datasets, AI validation datasets, and AI test datasets.

Example flows for dataset creation and supervised learning are shown in Figure 11–1 and Figure 11–2.

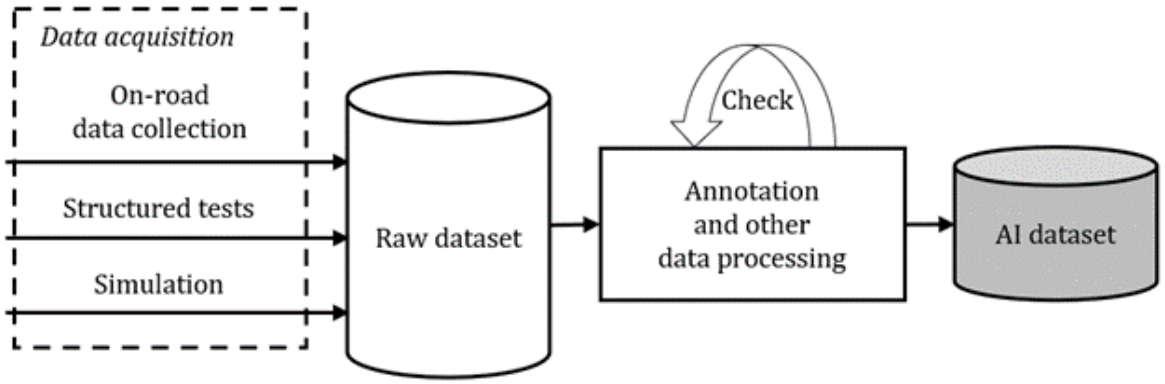


Figure 11–1 — An example dataset creation flow

© ISO 2024 – All rights reserved

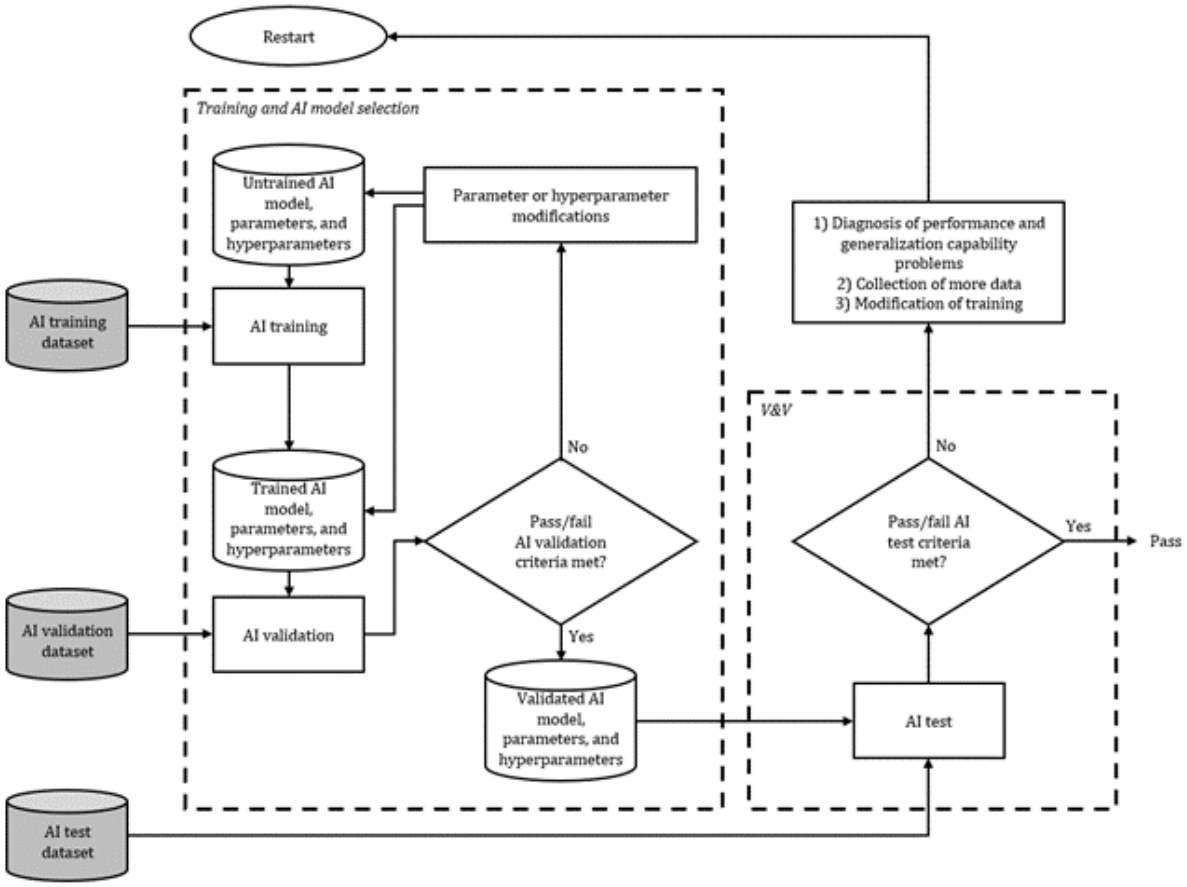


Figure 11–2 — An example supervised learning flow

The AI training dataset and AI validation dataset are used in the iterative AI training process of an AI model. The AI training dataset is input into the AI model while optimization is performed on its parameters and hyperparameters based on the AI model's performance. This proceeds until the predetermined AI training pass/fail criteria are reached. The AI validation dataset is then input into the AI model, and the AI model is evaluated against the AI validation pass/fail criteria. If the results are not satisfactory, hyperparameters of the AI model are refined and the AI training process is repeated.

Once the AI training is completed, the AI model is evaluated with the AI test dataset using the AI test pass/fail criteria as part of the verification and validation activities. If the verification or the validation fail, the process is continued after more data are collected and/or training is modified.

11.4.2 Reference dataset lifecycle

A typical dataset lifecycle describes the set of data-related activities carried out during the entire lifecycle of AI system development, including after deployment. The lifecycle serves as a means to manage the datasets and supports the realization of the AI safety requirements (and ultimately the safety requirements of the encompassing system).

A dataset lifecycle can consist of the following phases:

— dataset safety analysis;

— dataset requirements development;

— dataset design;

© ISO 2024 – All rights reserved

— dataset implementation;

— dataset verification;

— dataset validation;

— dataset maintenance.

A dataset lifecycle is created for the AI training, AI validation, and AI test datasets used in the AI system development (an individual dataset lifecycle can be created for each dataset role, if appropriate).

A dataset lifecycle can be aligned with or defined as part of the dataset creation and management activities at the level of the encompassing system, since system-level validation typically also relies on datasets.

Figure 11–3 provides an example dataset lifecycle based on the traditional V-model of development. Some of the salient features of the lifecycle are traceability of AI safety requirements to the dataset requirements (which impact the dataset's design and implementation) and an iterative workflow that extends into operation, where new data can influence dataset revision.

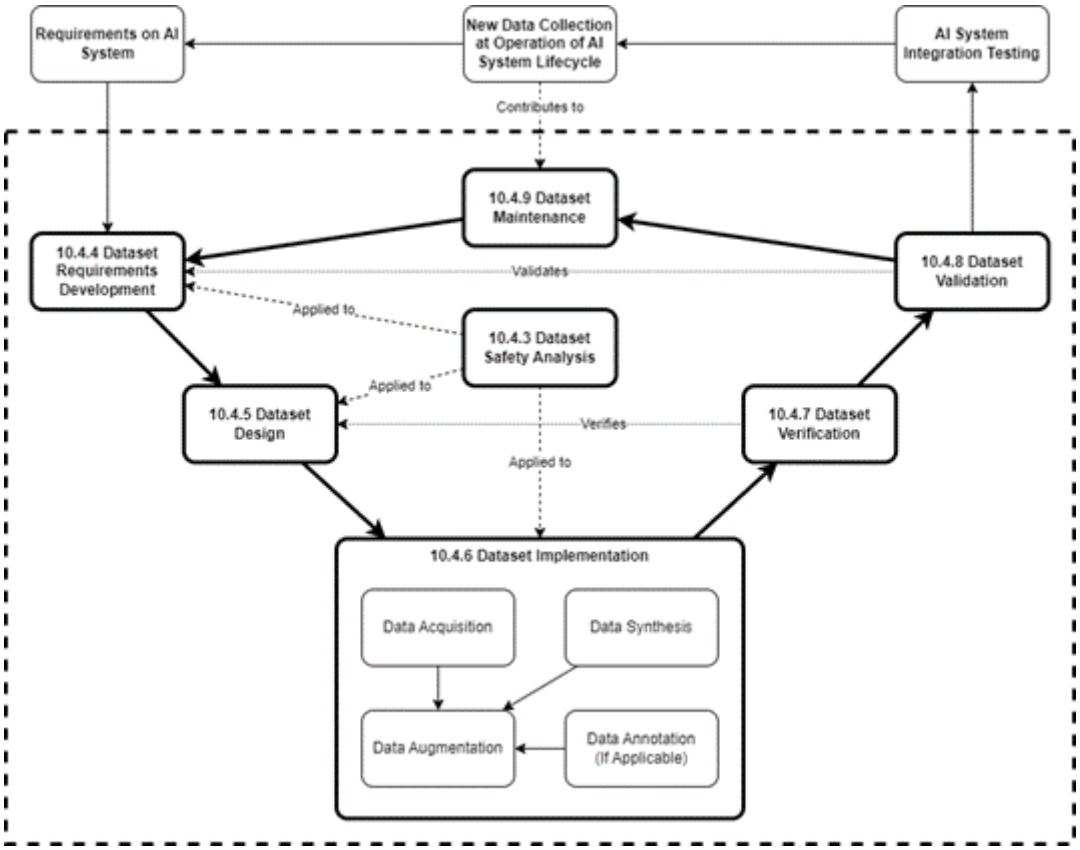


Figure 11–3 — Dataset lifecycle model

Clauses 11.4.3 through 11.4.9 discuss each dataset lifecycle phase in more detail.

© ISO 2024 – All rights reserved

11.4.3 Dataset safety analysis

11.4.3.1 General considerations

Dataset safety analyses focus on identifying safety-relevant dataset insufficiencies. When these dataset insufficiencies have been examined and the causes and consequences of such dataset insufficiencies have been identified (including the risks at the AI system and encompassing system), that information is fed as inputs to the dataset requirements development, dataset design, and dataset implementation to realize:

— countermeasures to prevent or mitigate dataset insufficiencies;

— metrics to judge achievement of the dataset insufficiency avoidance.

Depending on the dataset lifecycle phase, different approaches to dataset safety analyses can be used as outlined below.

Dataset requirements development phase:

— A *guideword-based approach* (such as that applied in HAZOPs) can be used to identify how dataset insufficiencies impact the safety of the AI system. Using this approach, one can determine what characteristics of the dataset(s) lead to the AI system performing a function incorrectly, seldom, too often, too little, too early, or too late.

— A *qualitative risk analysis approach* can be used to define the rigor applied in the dataset lifecycle to avoid dataset insufficiencies. Such an approach is analogous to the application of HARAs to determine ASIL in ISO 26262, and it considers:

  — the severity of the outcome associated with dataset insufficiencies (including the risks at the AI system and encompassing system);

  — the likelihood of the outcome associated with dataset insufficiencies;

  — existing countermeasures that can prevent or mitigate dataset insufficiencies;

  — additional countermeasures that can be applied to prevent or mitigate the dataset insufficiencies, and the degree of rigor with which such countermeasures can be applied.

Generally, more significant risks at the AI system and encompassing system can be associated with a qualitatively higher assurance level and hence a greater rigor in the robustness of the countermeasures that can be applied.

Last, a *residual risk analysis approach* can be applied after the impact of the countermeasures in risk reduction has been considered.

Dataset design phase: Various *deductive and inductive analysis approaches* can be conducted considering the proposed dataset design and can generate additional countermeasures that were not originally introduced at the dataset requirements development phase.

Dataset implementation phase:A *process failure mode and effects analysis(PFMEA)* approach can be employed to identify potential issues in the processes, methods, and tools of the data preparation and labelling and link these issues with the dataset insufficiencies and/or violations of the AI safety requirements (see Clause 9 of this document and ISO 26262-9).

EXAMPLE An AI system classifies in-path objects for alert and trajectory planning purposes. The AI system uses a colour image from a camera mounted on the windshield, and the AI training data, in particular, is hand-labelled. A PFMEA raises issues that can generate safety-relevant dataset insufficiencies such as:

— using images that were collected from the same camera on a different vehicle with different mounting and calibrations (data reuse impact);

© ISO 2024 – All rights reserved

— using images that were collected 10 years ago, even though there has been a considerable change in the types of vehicles that are on the road in the domain of interest (data ageing impact);

— using images that were collected from one region even though the system is targeted to operate in multiple regions with varying types of in-path object behaviour (data bias impact);

— using images that have not been labelled in a standard and correct manner (labelling inconsistency impact).

A summary of outputs of the dataset analyses serves as evidence to demonstrate that safety-relevant dataset insufficiencies are prevented or sufficiently mitigated. This summary can motivate or reference further artefacts (e.g., dataset tool qualification plan).

11.4.3.2 Dataset-related safety properties

Dataset insufficiencies are insufficiencies of the dataset regarding data-related safety properties under consideration. Examples of these data-related safety properties are given in Table 11–1, and they encompass both general properties and properties specific to AI applications.

Table 11–1 — Examples of data-related safety properties

Property	Definition
Accuracy	The data correspond to their source with respect to semantical representation and interpretation.
Completeness	The data elements (including metadata) are populated and the data have defined coverage of the input space, safety-relevant cases, and plausible data perturbations.
Correctness (or fidelity)	The data correspond to the phenomenon they intend to capture and include features and metadata which help to characterize the phenomenon.
Independence of datasets	The datasets sufficiently avoid leakage of information amongst themselves with respect to data sources and the methods used to capture, gather, generate, and process the data.
Integrity	The data are not altered by natural phenomenon (e.g., noise) or intentional action (e.g., usage of lossy data compression without consideration of impact to model, poisoning).
Representativeness	The distribution of data corresponds to the information in the environment of the phenomenon to be captured; it is free of biases.
Temporality	The data gives sufficient consideration to time-based characteristics (e.g., timeliness, ageing, lifetime, time contributing to distribution shift).
Traceability	The derivation of the data from their origin (including information on how they were captured, gathered, generated, and processed) is demonstrated.
Verifiability	The data include sufficient features to be amenable for verification as prescribed by their requirements and properties.

NOTE 1 ISO/IEC CD 5259-1 and the SCSC Data Safety Guidance Version 3.2 detail additional data-related safety properties (e.g., portability, understandability, auditability).

© ISO 2024 – All rights reserved

NOTE 2 Properties might not necessarily be mutually exclusive (e.g., a well-known property that is not listed is independence and identical distribution, or IID, which is covered by the correctness, completeness, and independence properties).

Regarding a dataset insufficiency due to lack of independence of datasets, independence between the AI training and AI validation datasets supports detecting overfitting. Though not required, the K-fold cross validation technique can support this as the AI training and AI validation datasets are independent in each of the folds. Independence between the AI training and AI test datasets, on the other hand, supports providing a reliable statistical estimation of the residual risk of the trained AI component.

Regarding a dataset insufficiency due to lack of representativeness, biases can manifest in different forms. Human cognitive biases impact how engineering decisions are made and how datasets are sampled. Non-human cognitive biases (such as sensors failing during data collection) result in systematic dataset insufficiencies. More information on these forms can be found in Clause 6 of ISO/IEC TR 24027.

Generally, a bias in the AI training dataset impacts the performance of the AI system and is unwanted. However, intended biases can be used as a design measure to put the AI training focus on some important but rare features critical to the safety of the AI system (e.g., an AI training dataset with a higher occurrence rate of corner cases can be used to complement an AI test dataset based on the real-world distribution). This design measure can still be insufficient to capture the true variability of rare safety-relevant cases, however.

Specific examples of dataset insufficiencies and the potential actions to avoid such dataset insufficiencies can be found in Table 11–2.

Table 11–2 — Examples of dataset insufficiencies

Property	Dataset Insufficiency Example(s)	Potential Actions to Avoid Insufficiency
Accuracy	The resolution of camera images is not sufficient according to expected AI model inputs for object detection. An AI system operates with sensors detecting certain type of obstacles at a given distance range and high speed, but the camera used is not adapted for the range and speed, yielding blurry images. The mesh used in LiDAR imaging of objects is not fine enough (number of points, spacing) to properly detect target obstacles.	— Selection of source sensors appropriate for the input space and application; — Inspection of manually labelled data.
Completeness	Few images have obstacles close to the camera in a dataset for obstacle detection. No night images are in the dataset even though the input space includes nighttime. Perturbations like noise, brightening, darkening, vibration, rotation, turbulence, blurring, blooming, smear, and interference are not reflected in the dataset. An AI system for traffic signal identification is trained with a dataset does not contain data elements that have all of the possible variations of traffic signal shape,	— Investigation of general use cases; — Calculation of distribution of the data and verification that the data cover the input space; — Collection of data from different geographical experts' perturbations on data that represent realities within the input space; — Addition of data through selection, generation, augmentation, or synthesis if there are gaps identified (e.g., by

© ISO 2024 – All rights reserved

Property	Dataset Insufficiency Example(s)	Potential Actions to Avoid Insufficiency
	height, positions, etc. outputted by the AI system. Missing information on the location of captured data does not allow one to analyse the geographical distribution of data and can cause undetected bias.	analysis such as examination of saliency maps, training, and testing); — Monitoring and collection of new or changing items within input space; — Corner and edge case collection.
Correctness (or fidelity)	Annotators manually create bounding boxes around objects inconsistently, which leads to object size per scenario to be calculated differently. No distinction has been made between a motorcycle and its rider in an image label, though this is relevant for the driving task that the AI system performs. A scene is marked as rain by an annotator although it is in snow. LiDAR provides ground truth for a camera outputting distance to an object. The vehicle collecting data drives through rain, which causes the ground truth to be noisier than in nominal conditions. Adhesive body markers provide ground truth for a driver monitoring system outputting head position, and the markers shift during the data collections.	— Characterization of the essential features of the target phenomenon; — Determination of the adequacy of the sensors to detect, observe, and capture the phenomenon; — Redundancy of sensors.
Independence of datasets	One frame in a sequence is in the AI test dataset and the next frame is in the AI training dataset. Due to frame rate, the frames are nearly identical. One frame of a certain geo-position is in the AI test dataset and another taken later at the same geo-position is in the AI training dataset. For object detection, this can compromise the independence of the datasets. All datasets used for an AI model development come from the same exact environment (e.g., same city street, time intervals, weather conditions, and traffic load). All datasets are collected relying upon the same means (e.g., only one sensor or database).	— Use of data management system; — Use of different sources of data; — Separation of the teams preparing the different datasets; — Use of different technical means for data capturing, e.g. different sensors, vendors, and brands; — Deployment of different processes/methods for dataset creation, e.g. applying two algorithms for data sample generation.

© ISO 2024 – All rights reserved

Property	Dataset Insufficiency Example(s)	Potential Actions to Avoid Insufficiency
	The task for dataset creation is always based upon the same technique, algorithm, or parameters and is conducted by the same person.	
Integrity	Corruption of hardware storage introduces error(s) in the dataset. Failure of database memory introduces error(s) in the dataset. Untrained/careless user inadvertently introduces inconsistent data element (e.g., altered label). Error is introduced during dataset processing/manipulation due to transfer over lossy channel.	— General inspection; — Analysis of robustness to adversarial attacks on the dataset (e.g., random erasing, corruption); — Standard access controls (e.g., authorized users with passwords, denial of service protection techniques); — Built-in features for integrity checks in databases and other data storage; — Inclusion of integrity check codes (e.g., CRC, checksum, hash) for storage and transfer over lossy channels.
Representativeness	An AI system for driver monitoring is going to be used in a region where drivers have an even distribution from 20 years old to 100 years old, but the dataset does not contain drivers above 80 years old. AI datasets collected by a heavy truck fleet can have geospatial bias for an AI system intended to perceive aspects of roadways with weight limits. Synthetic data do not capture have differences with real-world data to which the AI system is sensitive, like shadows in images. Real-world data have been captured with wrong sensor parameters, resulting in variances to which the AI system is sensitive. Sensor data have disturbances due to a bug on the camera.	— Analysis and comparison of the theoretical and experimental distributions of the phenomenon; — Distributional drift analysis.
Temporality	COVID-19 induced change in distribution of people wearing face masks, but the dataset for a driver monitoring system does not consider this.	— Inclusion of metadata containing details like time of creation and validity; — Usage of version control for the dataset.

© ISO 2024 – All rights reserved

Property	Dataset Insufficiency Example(s)	Potential Actions to Avoid Insufficiency
Traceability	An image lacking information on its source appears to be complete under simple visual inspection and is integrated into a dataset. Two datasets containing the same category of data are integrated into a single dataset without their metadata and attributes. The original datasets and their metadata are deleted. Data samples that were randomly selected for a training dataset decrease performance of the AI model due to those samples being corner cases. The samples are subsequently removed from the training dataset, but their metadata do not have an attribute to label them as corner cases. A new optimization algorithm is applied to datasets without properly tracing its application in the data management process. The optimized datasets are still used as replacement of the older ones.	— Use of data management system; — Updating of data management process to account for all tasks impacting datasets; — Creation and inclusion of appropriate and sufficient metadata to the data element collected and synthesized.
Verifiability	A framework that relies upon a random algorithm generates data samples that violate a safety indicator in a simulation run. Without sufficient mechanisms for reproducing the same run, the violation is likely unverifiable. No checksum, CRC, or hash mechanisms were activated at any time in the data management process, and the datasets cannot be integrity-checked. Dataset images cover a certain period of the year (e.g., autumn and winter), but the camera was not correctly configured with the actual date and the respective metadata is unreliable. — Different camera vendors and brands were used to ensure independence of data. However, all images were mixed and processed together, and the technical means were not recorded.	— Ensuring reproducibility of data generation; — Mechanisms to allow verification of data properties, e.g. built-in features in databases for integrity checks; — Discarding of datasets with unreliable or insufficient metadata; — Manual analysis; — Use of statistical sampling methods.

© ISO 2024 – All rights reserved

11.4.4 Dataset requirements development

The dataset requirements development follows the activity flow below, assuming that the method specified in ISO 26262-3 regarding item definition and the method specified in ISO 21448 Clause 7 regarding triggering conditions have been followed:

a) comprehension of the AI system;

b) dataset safety analysis;

c) dataset requirements formulation;

d) dataset requirements quality assurance.

The comprehension of the AI system activity of the dataset requirements development focuses on understanding the intended functionality of the AI system, including:

— the AI safety requirements, from Clause 9;

— the input space definition, from Clause 9.

The dataset safety analysis activity is performed in alignment with the guidance in Clause 11.4.3, and the outputs are fed into the dataset requirements formulation.

The dataset requirements formulation activity focuses on formulating the dataset requirements that mitigate the risks associated with the output of the AI system. It specifies:

— the logistical aspects, addressing the following items at minimum:

  — where the dataset is stored;

  — who has access to the dataset, what type of access they have, and when they have access, including consideration given to ensure that this dataset is safe from unintended editing;

  — how the dataset is version controlled and how changes are tracked;

  — requirements on the verification and validation processes to be employed to ensure that the data within the dataset is correct and appropriate for usage;

  — how stakeholders can report known vulnerabilities, risks, or biases in the data and/or dataset during any of the dataset life cycle phases.

— the technical aspects, addressing the following items at minimum:

  — size of the dataset;

  — format of the data within the dataset, including what syntactic and semantic parameters describe the data and what the format for labelling is;

  — boundaries of the data within the dataset (driven by both ground truth and design decisions);

  — dataset's role (AI training, AI validation, or AI test) and what ensures that it is sufficient for its given role, including limitations on how many times it can be used (to avoid overfitting);

  — constraints affecting creation of the dataset (e.g., region-specific data privacy regulations);

  — mitigations for the different manifestations of dataset insufficiencies detailed in Clause 11.4.3;

© ISO 2024 – All rights reserved

2317 — methods to prevent undetected data failures.

2318 Finally, the dataset requirements quality assurance activity focuses on ensuring that the dataset requirements
2319 follow the criteria given in ISO 26262-8, Clause 6. Requirements are:

2320 — traceable to the AI safety requirements;

2321 — updatable and maintainable upon a change to the encompassing system, the AI system, or input space;

2322 — updatable upon exposure of an insufficiency in the AI system due to the discovery of new safety-relevant
2323 scenarios or other triggers.

2324 **Table 11–3 — Further considerations for requirements development**

Requirements Topic	Considerations
Input space	For an ADS, “input space” is equivalent to “operational design domain”. An input space can also include the user demographic and user-driven parameters (e.g., a driver monitoring system can have an input space that includes drivers with face paint).
Dataset's role	Regarding the AI test dataset role, the AI test dataset is used specifically in the AI test verification part of the process, which can be performed as part of the vehicle-level verification and validation. Ideally, a large AI test dataset is used to uncover overfitting. In case of repeated performance measurements on the same AI test dataset, the statistical validity of the test results can be threatened by implicitly optimizing towards the AI test dataset, and the dataset requirements can include countermeasures to address this. Example countermeasures include the following: — Employing multiple independent AI test datasets for use across different iterations of AI system development; — Restricting access to the AI test dataset such that KPIs measured on a random subset of the dataset are returned instead of detailed results. Datasets can support another role – providing a means to monitor the input space while the AI system is in operation. Clause 11.4.9covers this use case in more detail.
Boundaries of data	An example of setting boundaries is a situation in which a data element for a particular dataset has the following format: <Time>, <Day>, <Traffic heaviness>. Time is bounded between 00:00 and 23:59, inclusive, day is bounded within the [Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday] enumerations, and traffic heaviness is bounded within 0 and 100%, inclusive.
Constraints affecting creation	An example of a constraint is a situation where a region in which a vehicle is going to operate has restrictions disallowing capturing images of individuals. This can

© ISO 2024 – All rights reserved

Requirements Topic	Considerations
	introduce an unwanted bias if the AI system's function is to classify pedestrians based on the images.
Traceability	Traceability from dataset requirements to AI safety requirements (including those which have been generated from consideration of the input space (refined)) can be evidence that the AI safety requirements have been sufficiently considered. An example of this is as follows: — An AI system has an AI safety requirement that specifies the acceptable false positive rates for the function that the AI system performs in two different weather conditions: sunny and rainy. — The AI training dataset has dataset requirements that specify how much of its data are in sunny conditions and how much of its data are in rainy conditions. Similar dataset requirements are created for the other datasets. — These dataset requirements link to the AI safety requirement.

11.4.5 Dataset design

The dataset design outlines details on:

— data elements that are collected physically, created synthetically, and/or created through augmentation (and how augmentation is applied in dataset generation and on-the-fly in the AI training pipeline, if applicable);

— which aspects of the data elements comprise the core data;

— the metadata, including any ground truth data associated with objects of interest within the data elements (also known as the labelling specification for supervised learning);

— the operations to be performed on the dataset (e.g., filtering of irrelevant or invalid data, dimensionality reduction, de-identification of data for data privacy purposes, normalization of the data with respect to appropriate metadata parameters, etc.);

— any mechanisms to be realized for monitoring the distribution shift in the input data during operation and collecting additional data items for subsequent revision of the dataset.

The details outlined in the dataset design are to be documented and subjected to analysis to ensure that they preserve the AI safety requirements and the dataset requirements.

11.4.5.1 Creation of data elements and identification of core data

There are three main approaches used for creating data elements in a dataset:

1) Physical collection of data elements: Data elements in the dataset are directly obtained using either the sensors used in the encompassing system or their surrogates. In the case of surrogate sensors, a gap analysis is done to assess the effects of the difference with the native sensors, and countermeasures are taken to contain the effects.

© ISO 2024 – All rights reserved

2) Synthetic creation of data elements: Certain aspects of the input space might not have been captured during data collection, and various simulation tools and specialized machine learning methods like generative adversarial networks can be employed to create additional data elements that capture those aspects. The adequacy of the synthetic data elements is considered.

3) Augmentation of physically-created data elements: A physically-created set of data elements are augmented to create a new set of data elements that have parameters altered to include perturbations such as noise, brightening, darkening, vibration, rotation, turbulence, blurring, blooming, smear, and interference. If augmented data are used for testing and if the data even partially replace real-world tests, the adequacy of the augmentation is considered.

EXAMPLE For computer-vision based data, the means of altering can include color space transformation, background modification, superposition of multiple images, flipping, scaling, translation, and random cropping.

The aspects of these data elements that serve as the inputs to the AI function during runtime operation of the AI function (e.g.. the raw image data from a camera sensor, a 3-dimensional 128 x 128 x 3 array of RGB pixel values, for an AI system built on computer vision) are the core data.

11.4.5.2 Design of metadata for data elements and datasets

The metadata associated with a data element provides valuable information about the data element that can be used during training, analysis, and verification. For a supervisory ML system, the metadata includes the ground truth or the label used during the training process. The data type, structure, and range of values that the labels assume are also identified during the design phase, and they conform to the functionality of the AI system and meet the dataset requirements.

EXAMPLE 1 An AI system used to classify objects for an automatic emergency braking system depends on an AI training dataset for which the data elements have metadata containing the class of the object (e.g. car, truck, person), the height and width of the object (the bounding box), and the distance of the object from the subject vehicle.

Metadata can also be associated with a dataset as a whole. The metadata associated with a dataset serves to support analysis, verification, and validation of the dataset, and can contain the following:

— details on how the dataset was created, e.g., physical collection, details of sensor devices, synthetic methods and tools that were used for generation, augmentation, etc.;

— statistics of syntactic and semantic parameters of data elements in the dataset;

— information that relates the data elements in the dataset to the AI system and the encompassing system, input space, object/event detection and response, dynamic driving tasks, edge cases, etc.

EXAMPLE 2 An image dataset can be defined to be such that 10% of its data elements contain traffic signals in the top left corner. In the assessment of completeness, the range of values of various parameters and their combinations can be useful.

11.4.6 Dataset implementation

The activities in the dataset implementation phase realize a concrete dataset based on the dataset requirements and dataset design, and they include:

— defining the processes, methods, and tools to prepare a given dataset (e.g., physical, synthetic, and/or augmented data generation, cleaning as covered in Clause 10.4.5, etc.);

— preparing the dataset;

— defining the processes, methods, and tools for labelling the dataset;

© ISO 2024 – All rights reserved

— labelling the dataset.

NOTE ISO/IEC 23053 identifies the data preparation step as a tool in the development of an AI system. ISO/IEC CD 5259 defines a data quality framework that can be used as guidance during this phase, and ISO 24368 provides guidance on having processes in place for stakeholders to disclose/report known vulnerabilities, risks, or biases associated with the dataset preparation and labelling, which can then be fed into a dataset safety analysis.

Regarding labelling in particular, it is often a human-labour intensive process involving a label supplier, and label quality tests and audits are applied. The nature and extent of these tests and audits is commensurate with the complexity of the inputs and outputs being labelled.

EXAMPLE 1 If the data to be used for an AI system development is LiDAR data involving point clouds, the labelling to identify objects in the inputs could be complex and error-prone. To address this, the labelling process can employ multiple levels of human involvement, e.g., labellers, reviewers, and auditors, and possibly also semi-automated and automated label quality tests and plausibility checks.

Additionally, for labelling, consideration is given to how any ground truth is obtained to ensure that any instrumentation used to obtain the ground truth does not interfere with how the data are represented.

EXAMPLE 2 Body markers being used to provide ground truth positioning of a person for a camera-enabled driver monitoring system can cause interference since the body markers will likely not be a part of the real-world operation distribution set.

As part of the dataset implementation activities, the details of records created during the preparation and labelling are documented as inputs for dataset safety analysis and dataset verification activities. The coverage of the input and output spaces and the statistical distribution of the datasets are also recorded.

11.4.7 Dataset verification

Dataset verification applies to the dataset under evaluation, and its purpose is to confirm that the dataset has been developed correctly. It comprises product verification complemented by process verification:

— product verification:

  — determining the consistency and correctness of information in a data element;

  — determining the consistency and correctness of information at the dataset and metadata levels, e.g., lack of outliers, missing data elements, duplicates, wrong data types, etc.;

  — verifying the conformance of the dataset against dataset requirements, e.g., metrics on distribution of parameters of the dataset, extreme values and edge cases of parameters and their combinations, noise characteristics, independence between datasets, etc.;

— process verification:

  — checking that the design and implementation phases are performed correctly;

  — checking the correctness of the processes, methods, and tools used to create the dataset and its metadata (including any other AI systems involved in ground truth labelling).

NOTE Product verification can be done either manually or using automated tools, depending upon the type of checking that is involved. For instance, verification of the information about the sensing device used for data collection can require manual inspection, while that of the ground truth label can employ running automated software. Checking the correctness of all of the data elements in a dataset can be impractical, so this can be done using statistical sampling approaches.

EXAMPLE For an AI system that is expected to learn certain high-level concepts (e.g., pedestrians, vulnerable road users, traffic signals etc.), the dataset is required to have a sufficient number of data elements containing these concepts

© ISO 2024 – All rights reserved

so that they can be learned. The AI system metric is that the AI system perceives vulnerable road users under certain lighting conditions with an accuracy of X%. This AI system metric relates to dataset requirements that Y% of the training dataset and Z% of the AI test dataset contain vulnerable road users under those lighting conditions. If these dataset metrics are not met, the dataset is enhanced with additional data elements.

Dataset verification is repeated every time dataset requirements are added or refined. The details of the verification carried out are documented as evidence for the verification of the dataset.

11.4.8 Dataset validation

Dataset validation ensures the correctness of the dataset requirements from the dataset requirements phase, i.e., if the correct dataset and data elements are developed for the AI system with the desired safety properties and if they reflect a correct translation of the AI safety requirements.

There are two approaches to dataset validation activities which can be applied together or individually:

a) requirement conformance, which involves checking that the derived dataset requirements meet the expected objectives of the dataset;

b) integration testing, which involves checking that the AI system developed using the dataset(s) (i.e., trained and tested with the dataset(s)) meets the AI safety requirements.

For the first approach, the expected objectives of the dataset are adequately articulated at the AI system development phase and handed over to the dataset development team. Often, the expected objectives are in terms of use cases and edge cases of the AI system, and checking that these are met is done by examining and reviewing the dataset requirements. As part of this evaluation, the consistency and completeness of the dataset requirements can be assessed.

In addition, requirement conformance can be carried out by examining the AI safety requirements. Every safety requirement that has a potential impact on a dataset is covered by one or more dataset requirements (although there can be dataset requirements that do not trace to an AI safety requirement). Requirement conformance involves checking that a correct and desired traceability exists, and can be carried out manually and/or using some automated support of analysis of the requirements.

The second approach to dataset validation is integration testing. In this approach, the AI system is derived or revised using the dataset, and the resulting AI system is verified against its requirements, with the additional objective of the AI system verification being to check that the right dataset was developed. Any failure in AI system verification can be traced to deficiencies in the dataset (and subsequently accounted for in the dataset requirements or design) or to other issues, like aninadequate AI system architecture or an inadequate training process.

The dataset validation phase results in evidence describing the relationship of dataset requirements to the AI safety requirements and summarizing the review results

11.4.9 Dataset maintenance

Dataset maintenance refers to the set of activities that ensure that a dataset is up-to-date and compliant with the dataset requirements. These activities are carried out across the entire dataset lifecycle.

Along the lines of ISO 26262-8, the dataset maintenance activities can include:

— configurations of the dataset, including what they are, what they do, and how they are managed;

— management of dataset resources, tools, repositories, access rights, and timelines;

— change management of datasets including, what triggers changes to the dataset;

— retiterment and decommisioning of dataset and their elements.

© ISO 2024 – All rights reserved

NOTE Guidance for retirement and decommissioning are available in ISO/IEC DIS 5338 and ISO/IEC DIS 8183.

Dataset maintenance activities include actions taken during operations (see Clause 14) and involve

— general field data collection and monitoring (e.g., monitoring the inputs to the AI system for conformation to the AI safety requirements and identifying data elements corresponding to safety-relevant edge cases encountered during operation);

— out-of-distribution (OOD) data identification, collection, and processing;

— AI system adaptation.

Table 11–4 — Examples of dataset maintenance activities

Sub-Category	Example(s)
Out-of-distribution (OOD) data identification, collection, and processing	When an AI system fails to detect certain objects, causing unwanted behaviours at the system level, data with those objects are collected and uploaded to fine-tune the AI model. An infrastructure feature that was in the AI system's input space has become obsolete. Data elements containing that feature are removed from the dataset. An AI system for driver monitoring performs sub-optimally when the driver is wearing a certain thickness of glasses. Data elements containing people wearing that thickness of glasses are added to the dataset.
AI system adaptation	When an AI system is deployed in another country, it might be expected to detect different signage and adjust system behaviour according to local laws and regulations. Data elements containing that signage are added to the dataset. An ADS that employs an AI system goes from operating on limited access highways to also operating on highways with at-grade crossings. Data elements containing at-grade crossings are added to the dataset.

11.5 Work products

11.5.1 Dataset lifecycle,resulting from 11.3.1 and11.3.2.

11.5.2 Evidence for the outputs of the defined phases of the dataset lifecycle, resulting from **11.3.3**.

11.5.3 Evidence for the safety analyses of the dataset, resulting from 11.3.4 and 11.3.5.

11.5.4 Dataset requirements specification, resulting from11.3.6 and11.3.7.

12 Verification and validation of AI system

12.1 Objectives

The objectives of this clause are:

a) to verify that the AI system fulfils its AI safety requirements;

b) to validate that the safety requirements allocated to the AI system are achieved when integrating into the encompassing system;

© ISO 2024 – All rights reserved

NOTE 1 This clause includes guidance for:

— the stand-alone performance analysis of the AI system itself;

— testing at the AI system and AI component level. Testing at the AI component level can include AI model, and pre-and post-processing elements.

NOTE 2 The term “validation” is typically used differently within the development of AIsystems than is used within safety standards and system safety engineering. In this document, the term “AI system safety validation” defined in*AI system safety validation* (3.1.21) is used.

12.2 Prerequisites and supporting information

The following information shall be available to complete the verification and validation activities associated with the corresponding phases of the AI safety lifecycle:

a) Safety requirements allocated to the AI system (from external sources);

b) AI safety requirements, from Clause 9;

c) Known insufficiencies of the AI system and the corresponding subdomains of the input space, from Clause 9;

d) Input space definition (refined), from Clause 9;

e) AI Component or AI System Architecture, from Clause 10;

f) Implemented AI component, fromClause 10;

g) Dataset lifecycle, from Clause 11;

h) Evidence for the outputs of the defined phases of the dataset lifecycle, from Clause 11;

i) Evidence for the safety analyses of the dataset, from Clause 11;

j) Dataset requirements specification, from Clause 11.

12.3 General requirements

12.3.1 The AI system shall be verified to provide evidence for:

a) compliance with the AI safety requirements;

b) confidence in the absence of unintended functionality and properties.

NOTE Confidence in the absence of unintended functionality and properties can be increased, for example, by following safety standards such as ISO 26262, ISO 21448 and this document during the development.

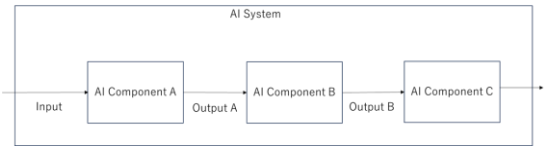
12.3.2 Testing of an AI system shall be performed on the AI components that can be tested stand-alone, and on the integrated AI system.

NOTE If an AI component cannot be tested stand-alone, the AI component is tested at a higher level of integration.Inthis case, the AI system integration strategy is devised to accommodate this testing.

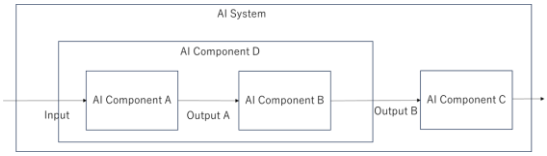
EXAMPLE 1 a)shows an AI system with three AI components. Assume AI component C can be tested stand-alone, but AI components A and B cannot. For example, suppose AI components A and B implement a convolutional neural network to propose bounding boxes and a non-max suppression algorithm to integrate them. In another example,

© ISO 2024 – All rights reserved

suppose AI components A and B implement a backbone/head (neural network), i.e. feature extraction such as vision transformers, shared with other AI systems and a task-specific neural network for this AI system. Output A is an intermediate value in these cases and cannot be tested stand-alone. However, AI components A and B can be tested if integrated together. In such a case, the integrated AI component is considered and tested stand-alone. 1 b)shows the integrated AI component (AI component D) within the AI system. Here, testing can be performed on AI components C and D and on the integrated AI system.



1 a) — An integrated AI system with 3 AI components



1 b) — AI components A and B are integrated so that the integrated AI component D can be tested stand-alone

12.3.3 Test cases for the verification of the AI components shall be derived using best practices for test case derivation including an appropriate combination of the methods listed in ISO 26262-6:2018, Clause 9, i.e. analysis of the requirements, generation and analysis of equivalent classes, analysis of boundary values and error guessing based on knowledge or experience.

EXAMPLE 1 Analysis of the requirements, including the required safety properties, might be used to select KPIs for the V&V activities.

EXAMPLE 2 Generation and analysis of equivalent classes might be suited to generate complete test sets for pre- and post-processing algorithms.

EXAMPLE 3 Error guessing based on knowledge or experience might be suited to identify yet unknown edge cases for testing.

EXAMPLE 4 Analysis of boundary values might be suited to generate complete test sets for pre- and post-processing algorithms.

NOTE 1 For machine learning, analysis of requirements relieson statistical tests to analyse whether the safety relevant performance requirements are met.

NOTE 2 If relevant, error guessing should include evaluation of known and potential triggering conditions, and evaluation of known and potential functional insufficiencies.

NOTE 3 The term “knowledge” should be interpreted broadly beyond human knowledge and can include knowledge automatically derived by an algorithm.

NOTE 4 For AI models used in perception modules of an autonomous driving vehicle, verification involves driving scenes in test datasets. One can refer to ISO 34502:2022 for more details on creating test scenarios for such models.

12.3.4 Each test case of an AI component shall include pass/fail criteria.

NOTE Pass/fail criteria can based on the formulation of the thresholds and parameters provided in the AI safety requirement allocated to the AI component, if applicable.

12.3.5 Test cases of an AI component shall adequately verify the AI safety requirements allocated to the AI component within the specified input space of the AI system.

© ISO 2024 – All rights reserved

NOTE 1 AI test quality and safety-aware AI testing are considered for these test cases. AI test quality refers to the necessity of rigorous testing of AI models that goes beyond any simple mean performance calculation with a single test dataset. Safety-aware testing refers to testing that uses safety-aware metrics and safety-relevant data points or subsets.

EXAMPLE 1 Test cases are designed to verify the AI model in terms of out-of-distribution performance and in-distribution performance on samples underrepresented in the training data.

EXAMPLE 2 Test cases reflect data points contributing to unsafe states of vehicles deductively enumerated by safety analysis such as design FMEA or FTA and inductively known from past products.

NOTE 2 For automated driving applications, the completeness and sufficiency of the test cases can be evaluated considering the acceptance criteria defined in ISO 21448.

NOTE 3 If the AI task is implemented by multiple AI models, the relevant sub-domain of the input space for each AI model is defined, e.g. one AI model could be used to explicitly identify vulnerable road users (VRU) while another could be used to explicitly identify traffic signs, resulting in a relevant input space subdomain VRU and traffic signs. The test cases would then be more focused on the relevant input space subdomains and less on the overall input space of the AI system.

12.3.6 The AI system integration approach shall specify the steps for integrating the individual AI components hierarchically into higher level AI components until the AI system is fully integrated.

12.3.7 The AI system integration shall be verified to provide evidence that the hierarchically integrated AI components, and the integrated AI system achieve:

a) compliance with the AI system architectural design in accordance with Clause 10;

b) satisfaction of the AI safety requirements.

12.3.8 AI system safety validation shall confirm that the safety requirements allocated to the AI system are fulfilled when the AI system is integrated into the encompassing system.

12.4 AI/ML specific challenges to verification and validation

AI systems, especially those developed using data-driven methods pose some unique challenges to verification and validation.

— They lack a precise statement of AI safety requirements. AI systems are often expected to identify and quantify human-interpretable high level semantic concepts like road objects, traffic signals, attendant driver, etc. Many of these concepts lack precise definitions and are difficult to capture in terms of mathematical descriptions.

— The inputs to an AI system are often versatile/diversified and from different sources, e.g. radar, LiDAR, camera etc. whose representation requires high dimensional objects with a very large range of values and satisfying complex and unknown constraints. Use of traditional input coverage methods would be incomplete or expensive.

— AI systems involving Deep Neural Networks employ complex architectures, especially for perception-based applications, e.g., Long Short-Term Memory (LSTM) networks, encoder-decoder networks and contain several layers with millions of parameters whose values are tuned during the training process. Verification, amounting to checking that these parameters have the desired values for optimum performance and to satisfy AI safety requirements, leads to scalability issues for many realistic applications.

— Training AI models involves the use of various heuristics for identifying parameter values to optimize appropriate cost functions. These heuristics could lead to locally optimum parameter values, without achieving a globally optimum solution (i.e. model generalization) that can lead to AI errors in the AI

© ISO 2024 – All rights reserved

system. The verification task involves checking that the computed parameters are adequate to satisfy the AI safetyrequirements which is a challenging problem.

— AI systems rely on a large dataset for their reliable performance. Demonstrating the validity and completeness of the verification dataset is a non-trivial task given the complexity and scope of the input space.

— The non-predictable erroneous behaviour in data-driven AI systems, for example, based on spurious correlations, limit the ability to predict performance based on a review of the training data or the implemented model.

NOTE This property is closely related to the property of robustness. This challenge can be further exacerbated by the lack of explainability of the trained function when using technologies such as deep neural networks.

— Limitation of structural coverage: Due to the lack of a detailed specification as well as the dependency on parameter values (e.g. weight in a NN) during execution, both black-box and white box coverage metrics have limited use when extrapolating the results of executed tests in evaluating performance over the entire input space.

— Stability of performance due to changes in the environment or the function. Small changes in the input space (e.g., one or two pixel values in an image input) can lead to, as yet undiscovered, AI errors in the function. Furthermore, changes to the function (due to re-training) can lead to an unpredictable impact on previously verified properties.

— An AI system while training with an inadequate number of examples can reach a local optimum that results in behaviors not aligned with the desired outcome.

12.5 Verification and validation of the AI system

12.5.1 Scope of verification and validation of the AI system

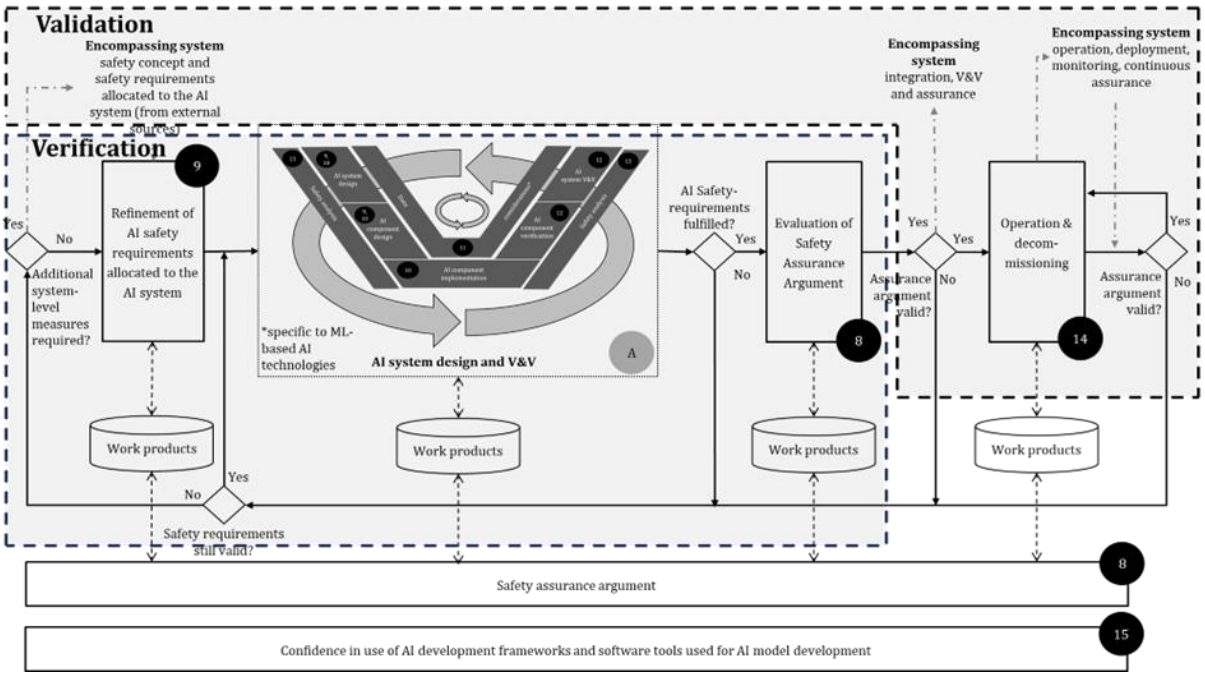


Figure 12–2 — Phases of verification and validation activities of the AI system

© ISO 2024 – All rights reserved

Figure 12–2 shows the phases where verification and validation activities of the AI system happen. Verification of the AI system is applicable to the following phases of the AI system safety lifecycle.

a) When defining the AI safety requirements (Clause 9), verification ensures that the AI safety requirements are correct, complete, and consistent with each other and with respect to the encompassing system technical safety concept and safety requirements. Verification of the AI safety requirements can be performed following ISO 26262-8, Clause 9 and ISO 21448.

b) During the development phase of the AI system, verification is conducted in different forms, as described below:

— During the design phase (Clause 10), verification is the evaluation of the work products, such as architectural design, models, or architectural measures, thus ensuring that they comply with the AI safety requirements for correctness, completeness, and consistency. Evaluation can be performed by methods such as review, simulation, or analysis. It is planned, specified, executed, and documented in a systematic manner following ISO 26262-8, Clause 9.

— In the data lifecycle (Clause 11), data is verified at each phase for sufficient correctness, consistency and completeness.

— In the test phase, verification of the AI system is the evaluation of the work products and elements within a test environment to ensure that they comply with the AI safety requirements. The tests are planned, specified, executed, evaluated and documented in a systematic manner.

Testing of an AI system is performed at different levels of system integration.

— AI component testing (described in Clause 12.5.2);

— Testing of the integrated AI system (described in Clause 12.5.4);

— Integration of the AI system with the encompassing system and AI system safety validation (described in Clause 12.5.7);

— Post-deployment validation (described in Clause 14).

In 12.5.2 to 12.5.7, some guidance on the verification and validation of AI systems is provided whilst addressing AI/ML specific challenges.

© ISO 2024 – All rights reserved

12.5.2 AI component testing

12.5.2.1 Testing workflow of an AI component

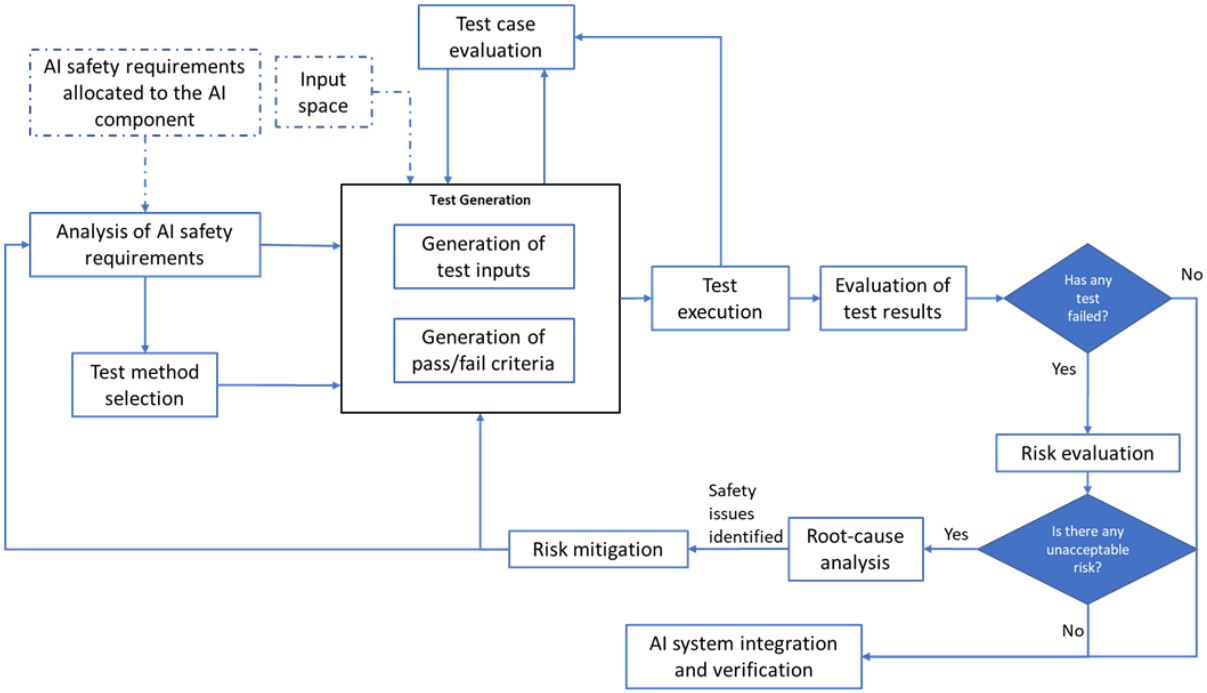


Figure 12–3 — Testing workflow of an AI component

Figure 12–3 shows a typical workflow for the testing of an AI component, based on the process presented in [20] for the testing of ML-based systems. Initially, the AI safety requirements allocated to the AI component are analysed to select the most appropriate combination of test methods. The AI safety requirements can also provide direct input to test case generation approaches which involve (automatically) deriving test cases for the AI component under test. The result of the analysis is a definition of expectations on the inputs, methods, environment and tools for performing the verification.

NOTE 1 Test planning of an AI component includes gathering information about related test methods, test evaluation criteria, such as test coverage or number of tests passed, etc..

NOTE 2 AI safety requirements include safety properties of the AI system and safety performance indicators.

This is followed by the formulation of the pass/fail criteria (test oracles) to be applied to the test results. These criteria can include a combination of coverage criteria (e.g. from the input space or from the AI model) as well as performance targets (e.g. "at most x% of inputs lead to an output with a deviation from the ground truth of no more than y%"), and safety-related KPIs.

Test inputs are sampled from the collected data or generated using simulation or other methods. See 12.5.2.2 and 11.4 for more details on test input generation. Test cases are evaluated to ensure that they sufficiently cover the scenario space considered, and also that they completely verify AI safety requirements within the input space of the AI system.

NOTE 3 Adequacy of the test cases for verification of the AI safety requirements can also be evaluated upon the execution of the tests. This is an ongoing area of research and the idea is similar to the structural coverage analysis for conventional software. See 12.5.2.3 for more details.

During the test execution, the AI component is run with the test inputs, and test results are generated. The test results are evaluated with respect to the defined test oracles.

© ISO 2024 – All rights reserved

NOTE 4 When evaluating the risk due to AI errors, one can consider the required target metrics and KPIs. It is possible that an AI safety requirement is not violated by the occurrence of a single AI error if the corresponding target metrics and KPIs (as defined in Clause 9) associated with the AI safety requirement are nevertheless met.

If a test fails, a safety analysis is conducted to evaluate the impact of the AI error(s) on safety. If the risk due to the failed test is deemed unacceptable, the root cause(s) of the AI error(s) are investigated. Then, depending on the potential root cause(s), appropriate mitigation measures are applied to reduce the risk. If the AI component is modified as a result of a risk mitigation measure, the AI component will be retested. The safety analysis (including risk evaluation, root cause analysis and identifying mitigation measures) will be discussed in more details inClause 13.

NOTE 5 It is assumed that at the stage of "Test Execution", the AI model is ready to be tested. For ML models, it means that the model is already trained and model parameters are set and the model is ready to be tested.

12.5.2.2 Test generation

Test generation involves identifying a test dataset that determines whether the AI system meets its AI safety requirements. The test dataset contains data elements which are representative of the inputs that the AI system receives in operation.

EXAMPLE An AI system used in a camera-based perception component of an automated driving system might receive road scenes as inputs which are dependent upon the ODD of the underlying feature, and the test input scenes would be required to have sufficient coverage over the entire ODD.

NOTE Test Generation for validation at the whole vehicle level is decribed in ISO 34502.

The AI safety requirements can include some requirements related to AI errors, and testing these requirements require generating edge cases based upon the input space definition. Applying methods for the systematic exploration of the input space can support an argument that both nominal and edge case conditions triggering AI errors are covered during testing. In general, there can be an expected distribution of nominal and edge cases over the input space. The test cases are representative of the input space and capable of uncovering critical AI errors.

In general, it is difficult to achieve the desired distribution for the test data when generating the data from actual measurements taken from the environment and using the target sensor set. Simulation environments can be used to generate additional (synthetic) test data to achieve therequired coverage and distribution over the input space. The input space can be described by a set of constraints on the input parameters and the desired coverage can be expressed and achieved using techniques such as Design of Experiments. When synthetic test data is used, the test evaluation activities include checking the validity of the generated test data.Due to the black box nature of certain ML methods (e.g. CNN), formal validation of the generated data is not possible. It is also not feasible to determine what features of the data the AI model is sensitive to (e.g. shadow, edge softness, blurs). Proving that insights gained on synthetic data can be transferred to real world data involves training multiple models on various composition of real and artificial data and comparing the performances.

12.5.2.3 Test case evaluation

When test cases are generated, they are evaluated to make sure they:

a) are appropriate and correct to evaluate the AI safety requirements allocated to the AI component, in particular regarding the expected outcome (the ground-truth);

b) adequately cover the AI safety requirements allocated to the AI component within the specified input space of the AI system;

c) effectively cover the scenario space considered.

© ISO 2024 – All rights reserved

In SOTIF, a triggering condition is always characterized by some range of parameters. For example, for a pedestrian wearing black clothing, there can be different interpretations, reflecting a range of RGB values that can be defined as black. Another example is the rain intensity, where for heavy rain, the intensity of precipitation is not a concrete value but instead included in a pre-regulated range. Therefore, the set of test cases ensures coverage of the range of values within the syntactic space that correspond to triggering conditions defined in terms of the semantic input space.

Once the input space has been characterized, methods such as combinatorial testing (see Clause12.5.3) can be used to perform such a check on relative completeness, where each relevant input dimension is partitioned into a set of finite valuations (e.g. for “fog”, further partition the intensity into three discrete classes “strong”, “moderate” and “low”), followed by ensuring all combinations being considered in the set of test cases.

NOTE Evaluation of test cases on the whole vehicle level is described in ISO 34505.

For covering the scenario space, for deep neural networks, a complementary method is to perform interpretability analysis (e.g. saliency maps) to understand the scenario or feature of why a particular neuron gets excited or suppressed. This can be used to define a coverage criterion to ensure the relative completeness of learned-feature combinations (e.g., neuron coverage and its variations or k-way combinatorial testing on neuron activations). However, achieving full coverage using such methods might not be possible, and methods to artificially ensure coverage can lead to generating random images that are never observed in practice.

12.5.3 Methods for testing the AI component

A suitable testing method of an AI component needs to consider, under available resources (e.g., time or computing power), both the breadth (to efficiently covering the input space) and depth (to enable effective issue/error detection) of testing. Practically, a portfolio of diversified methods is used for testing AI components. The following is an incomplete list of methods for testing AI models:

— **Statistical testing:** This method evaluates the achieved values of the metrics defined within the AI safety requirements and associated safety-related properties within a given confidence interval. Effective experimental design plays a crucial role in statistical inference (including statistical hypothesis testing) by safeguarding the validity and reliability of findings, thus preventing the production of spurious or distorted associations.

— **Data / scenario replay:** This method refers to collecting a set of scenarios (for example, recorded during test drives) and subsequently using the collected data to stimulate the AI model under test and evaluate the responses. Examples of these scenarios include known pre-crash scenarios from the National Highway Traffic Safety Administration (NHTSA) for motion planning [21]or WildDash for perception [22].

— **Random testing:** This method refers to test cases created based on randomly generated parameters from the input domain. For automotive, vision-based noise can include Gaussian noise or artificial occlusions.

— **Metamorphic testing**[23], [24]: This method refers to defining metamorphic relations to transform one test case into another. Metamorphic relations characterize the relationship between the change of input and the change of output.

EXAMPLE 1 The metamorphic relation might describe that when switching from daytime to nighttime while keeping the rest of the factors the same, the predicted bounding box size of the pedestrian should be the same.

— **K-way combinatorial testing[25]against pre-specified input space dimensions[26][27]**: This class of methods is introduced during the definition of test coverage. Dimensions of the input space are analysed from the input domain to identify equivalence classes in which a uniform behaviour of the system is expected. Then for any K dimensions, test all possible discrete combinations of those parameters on the AI model.

EXAMPLE 2 Consider a simple input domain is characterized using the following dimensions: time-of-day ∈ {daytime, evening, night, dawn}, weather ∈ {fine, cloudy, rainy, snow, fog}, and road-intersections ∈ {lane-diverging, lane-merging,

© ISO 2024 – All rights reserved

straight}. 2-way combinatorial testing needs to ensure that the set of collected test cases can cover, for any arbitrary two dimensions (e.g., <time-of-day, weather> or <time-of-day, road-intersections>), every pair of elements (e.g., <night, snow> or <dawn, lane-merging>) can be covered with a minimum amount of test cases (e.g., at least 1 test case). Using combinatorial testing,one can argue the relative completeness of testing efforts.

— **Boundary value testing or corner case testing:** This technique refers to testing boundary values of the input parameters. Typically, the parametrization of the boundaries of the input domain for complex tasks can only be accomplished approximately and with significant effort. Thus, testing AI systems at the boundaries of the input domain requires additional methods as compared to the non-AI case.

NOTE 1 The equivalence classes within the semantic input space might not correspond to the features within the syntactic input space learned by the AI model.

— **Gradient-based search methods or other open-box optimization-based testing methods:** This class of methods utilizes the knowledge of internal model parameters of the ML model to guide the generation of test cases. One can design optimization objectives such as creating erroneous prediction and, subsequently, utilize a given input's derivatives (gradient) to move towards the optimization objective. It is an open-box technique as the gradient is made available to the testing tool. Methods in adversarial perturbation (e.g., FGSM[28] or PGD [29]) utilize this principle.

— **Genetic algorithms or other closed-box optimization-based testing methods:** In contrast to open-box testing methods where the gradients are made available, closed-box optimization methods still try to perform the change of the input without utilizing the gradient information. Genetic algorithms use an initial population of test inputs and perform mutation of parameters to generate new test candidates. The next-generation of test cases are based on the pool of current-generation test cases that are best performers, with the definition of best characterized by the degree of violation. Other closed-box methods in the falsification of cyber-physical systems (e.g., simulated annealing[30] or Bayesian optimization [31]) use random samples to guess the gradient direction, in order to transform an input into another that can lead to undesired situations.

— **Probabilistic sampling-based test methods**[32]: These methods assume the availability of some prior belief about the distribution of AI errors within the input space. The areas with higher AI error distribution would then be sampled more often with the aim of finding triggering conditions of AI errors more efficiently.

— **Synthetic test case generation**: The synthetic test case generation allows generating edge cases that can be dangerous to be reproduced in physical world, as well as creating diversities in the scenarios being collected.

— **Testing based on expert knowledge**: Knowledge-driven testing refers to applying domain-specific know-how to create test cases, thereby checking if the model under analysis exhibits performance limitations that lead to safety concerns. For example, automotive perception component providers maintain a database of edge cases (e.g. jaywalking on a foggy night, pedestrian walking out of a pile of snow) that are considered challenging scenarios in products of the previous generation. These edge cases correspond to potential triggering conditions to the model under analysis.

— **Tests that analyze resource limitations (e.g. runtime):** This method covers activities such as ensuring that the model can be operated under the specified frequency (e.g., 10 FPS).

— **Robustness testing:** This method refers to considering the application's noise patterns or reasonable transformations and checking if the prediction subject to noise or transformation produces results consistent with the input without noise.

© ISO 2024 – All rights reserved

EXAMPLE 3 For an AI component working on audio data, robustness with respect to noise is of interest. In robustness testing, noisy audio signals are applied to the component, and it is verified whether the behaviour is as described in the requirements, e.g. that the performance does not fall below a certain threshold for noise up to a certain amplitude.

EXAMPLE 4 Autonomous vehicles rely on their perception capabilities to interact with the surrounding environment, which can be influenced by changes such as weather and lighting conditions. During robustness testing, it is important to consider adding perturbations to the test data across multiple dimensions simultaneously. For instance, in the case of image data captured by cameras, various types and intensities of weather conditions such as dusk and heavy rain can be introduced under different lighting conditions. This enables the evaluation of the system's accuracy in perceiving targets under challenging scenarios and assesses the model's adaptability to different combinations of disturbances. By subjecting the AI system to diverse and realistic environmental variations during testing, the effectiveness and robustness of the AI system in handling real-world conditions can be ensured.

— **Tests based on model analysis/review:** The performance on subsets of data is analysed to identify weak spots and/or lack of fairness. Subsets of data are identified where an AI component has weak performance. The errors are analysed to determine whether this is due to a systematic problem, e.g., low perception performance for bright frames because there were no bright frames in the AI training data set. Furthermore, test for potential assumed weaknesses of the model architecture fall into the class of test methods. For example, if a DNN for object detection is sensitive to rotated objects due to its architecture, this might motivate tests with rotated objects.

Apart from testing, one can also apply methods that make use of formal verification: Methods making use of formal verification can be categorised into those that are sound and complete or just sound.

— Exact methods (sound and complete) via specialized constraint solvers (e.g. [33]) or a reduction to mixed-integer linear programming or convex optimization.

— Sound methods based on methods such as abstract interpretation [34][35]. These types of methods can guarantee safety when the solver returns "safe". However, when the solver returns “unsafe” and provides a counterexample, the counterexample can be spurious due to over-approximating the state-space in the verification process.

NOTE 2 For formal verification of deep neural networks, beyond the issue of scalability, the lack of a precise specification and characterisation of the input space is one of the critical challenges[36] in the application of formal verification approaches to AI-components with high-dimensional input spaces (such as images). Therefore, some state-of-the-art approaches on image models, due to inability to mathematically characterize the input space, restrict the use of formal verification to the evaluation of robustness against perturbations over selected test samples.

12.5.4 AI system integration and verification

Based on ISO 26262-6:2018, Clause 10.2, software integration and verification refers to the activities where suitable integration levels and the interfaces between the software elements are verified according to the software architectural design. Moreover, ISO 26262-4:2018, Clause 7 discusses system and item integration and testing. ISO 21448 (Clause 10.6, Table 10) offers an additional list of methods for integrated-system verification. In principle, activities conducted in ISO 26262 and ISO 21448 can also be used to support the AI system integration and verification.

If two components that both contain AI components are integrated and if these AI components are not stochastically independent, test methods that address the statistical nature of the AI components are used to verify statistical properties of the composition. This can be done using the methods listed in 12.5.3. Stochastic independence, in this context, informally means that the correctness of the output of the first AI component does not influence the probability for a correct output of the second component. Formally, two events are stochastically independent if the probability for the occurrence of both events is equal to the product of the probability of occurrence of the individual events.

EXAMPLE Two object detection components, one based on camera, and one based on LiDAR, are both affected by occlusion of objects. If the component using the camera input does not detect an occluded object, it is more likely that the

© ISO 2024 – All rights reserved

component using the LiDAR input also does not detect it. Thus, the probability that both components do not detect a certain object is not the product of the probabilities that each component does not detect the object. They are not stochastically independent.

NOTE If statistical properties need to be verified after an integration step, statistical test methods should also be employed. In more detail, statistical properties of a component can be verified on component level. However, typically there are also statistical properties on higher integration levels or AI system level that are relevant. Often these cannot directly be taken over from component level because of the effect of other components.Thus, statistical verification can be deferred to higher integration levels or the system level.

12.5.5 Virtual testing vs physical testing

ISO/IEC TR 5469 provides a detailed discussion on virtual testing and physical testing for functional safety in AI systems. In addition, it provides guidance on how to assess the virtual test platforms.

This subclause briefly describes challenges with physical testing of AI systems and focuses on some of the advantages of virtual testing for AI systems in road vehicles.

NOTE Requirements for the usage of virtual test platforms in the context of validation of whole vehicle systems are described in ISO 34502:2022- 4.6.4.3.

12.5.5.1 Virtual testing

For AI systems operating within complex environments, there are challenges in physically testing an adequate range of use case conditions (e.g., weather effects, behaviour of other road users, etc.), and in particular to achieve a sufficient coverage of edge cases. In these particular situations, virtual testplatforms can be used to simulate all the desired variations.

NOTE 1 Edge cases can be defined as scenarios with very specific and rare conditions like for example extreme weather conditions, sun glare, specific environment conditions (erased road marking), etc.

Another capability of virtual testplatforms is the synthetic generation of datasets used for AI system development. Synthetic data generation allows for the generation of synthetic ground-truth information automatically from virtual test platforms. Indeed, within a simulation environment, not only the virtual image as seen by the sensor is simulated, but as an omniscient environment, simulation frameworks can also generate additional information (also known as ground-truth data) such as depth, object segmentation, object materials, bounding boxes or optical flow. Those data are important in the evaluation of performances of the AI models as it provides data as seen from a perfect sensor. Ground-truth data are challenging to obtain from real-world observations, and require in most cases a complicated, error-prone, and time-consuming manual process. Additionally, generating synthetic data sets allows for the training, validation and testingof AI models with independent data, which is key to avoid coincidental correlations, as stated in Clause 11.4.3.2.

NOTE 2 Independence between synthetically generated data is a complex subject and can be achieved by coverage analysis of the different environment parameters (daytime, weather conditions, sensors positioning, etc.), by using different sensor types among the available in the catalogues, by using different ground-truth generators, by variation of different simulation parameters (for example, scenarios length), by following independent processes when generating the synthetic data (for example, relying upon different methods or stakeholders to accomplish process tasks), etc.

As the dataset generated for AI training would be produced within a virtual environment, the entire generation workflow needs to be validated, and correlation with real data has to be made. To make sure we can rely on virtual datasets, comparisons between virtual and physical datasets have to be performed. Real world conditions, for instance, adverse weather, can be reproduced, tuned, and measured precisely in the laboratory (for example, see [37]). Once such data are collected, the same scenario and conditions can be set in the virtual environment for advanced correlation. It also helps understanding the gaps and domain of validity.

Moreover, the use of physics-based solvers on different disciplines (optics, electromagnetic, thermal, etc.) can be validated by independent certification bodies. This means being able to handle physics-based data as an

© ISO 2024 – All rights reserved

input (e.g. materials, emitters, sources, etc.), implement laws of physics within the solver but also generate the outputs that imitates the real conditions (e.g. spectral images, point clouds, range-doppler, etc.).

Using HiL for synthetic data validation

Hardware-in-the-Loop testing can be also used to test the accuracy of synthetic datasets ; it is one of the most accurate ways to correlate between real and virtual datasets. The sensors can be used to record scenes from the real world and thus, by reproducing and injecting the exact same scene from the virtual world into the real sensor, comparison between the real and the virtual dataset can be performed (by comparing the results of the HiL testing) as the only changing parameter is the dataset (sensor receivers and post-processing unit used in the real and virtual cases are the same).

12.5.6 Evaluation of the safety-related performance of the AI system

The performance of an AI system refers to the level of precision for prediction, the accuracy for classification, and the efficiency of an algorithm. Evaluation of the performance of an AI system is typically carried out with comparison of the system's output to the output from a benchmark, using a dataset which is proposed as, or has become, a standard dataset by which different solutions are evaluated. However, the performance of the AI system is "brittle" in the sense that the AI system that performs well has generally either been tailored to solve particular problems, or trained on specific set of data relating to the problems in a particular domain. Therefore, the lack of universally accepted or formalized criteria to assess the safety-related performance of the AI system poses an additional hurdle for widely adoption or utilization of an AI technology. This subclause aims to provide some guidance that fits into the framework of evaluation process of the safety-related performance of the AI system.

NOTE 1 AI models, particularly ML-based models, manifest the characteristics of statistical models. Traditional functional safety requires the system to be predictable, and hence the AI model's behaviour might be predictable in a probabilistic sense. It should be noted that predictability does not equate determinism, as it does in traditional software development. This implies that the AI system can contribute to a failure which is caused by software itself.

The following are examples of common causes of the negative impacts on safety-related performance due to AI errors.

— Inadequacy and uncertainty in the learning process: The learning process is instrumental for any ML-based system to generate accurate and reliable outputs. Insufficiencies in the training and test data, dynamically changing environments, and unpredictable intentions of road users, etc. can lead to unreliable learning results or misinterpretations by the AI system.

— Inappropriate cost function selection: The cost function is either less representative, or not affordable to re-evaluate in a consistent manner. This would result in negative side effects or reward hacking.

— Inappropriate metrics: using metrics not suitably matching the actual goals and priorities obscures the general system performance.

— Inconsistency between trained AI model and deployed AI model.

— Lack of benchmarks: Lack of universally adopted benchmarks which are reliable, transparent, standard and vendor-neutral results in performance differences between different parameters, even within the same application domain.

The applications of ML-based AI systems are usually categorized as follows:

— Regression, where the task is predicting a continuous quantity; and

— Classification, where the task is predicting a discrete class label.

Clearly and unambiguously defined metrics are required to evaluate the performance of an AI system, which in turn implies the safety level of an AI system (e.g., using performance indicator as a pass/fail criterion of the

© ISO 2024 – All rights reserved

system). A summary of the widely-adopted performance metrics for both categories are listed in Annex H. The metrics included in Annex H are by no means an exhaustive list of what industry is currently using. Other safety-related metrics can be derived, based on particular use cases and domain experts' knowledge and judgement, to evaluate the AI system from specific aspects of the system requirements.

NOTE 2 The performance metrics included in Annex Hare different from the loss functions. Loss functions are measures to quantify the model's performance during training process, while metrics are used to monitor and evaluate the performance of trained models in testing phase.

12.5.7 AI systemsafety validation

In contrast to verification, AI system safety validation refers to the activity of checking if the safety requirements allocated to the AI system (from the encompassing system) are met after the AI system is integrated into the encompassing system. AI system safety validation activities are usually done by the system integrator (e.g. OEM), where the validation target is defined separately. The AI system developer might need to support the activity.

For AI system safety validation, the individual methods listed in 12.5.3 can also be used. However, the focus is on the systematic exploration of all relevant scenarios within the input space and to examine abnormal situations. Systematic random testing by first discretizing the input space [38][39] is an example of such methods to argue relative completeness.

NOTE If the verification of the encompassing system admits the usage of virtual techniques like simulation, then the safety validation of the AI system, once integrated into the encompassing system, can also be based upon virtual techniques, for instance simulation can be conducted to systematically explore relevant scenarios and identify corner cases or abnormal situations.

For deep neural networks, AI system safety validation using field testing (e.g., by operating a fleet of autonomous driving vehicles) can be done with the assist of active learning methods or other methods for detecting out-of-distribution data. The underlying idea is that active learning methods try to infer if an input can be included in the training dataset by considering how different is this input with all existing training data.

As explained in Clause 9, in addition to the standard SOTIF-related AI safety requirements that directly address the performance targets, the workflow can introduce additional AI safety requirements that concretize AI safety-related properties (e.g. robustness, interpretability). Safety validation of the AI system also considers validating the appropriateness of these additionally introduced requirements. The purpose of the validation is to ensure that no insufficiencies of the specification exist. In particular, the activity ensures that the quantitative thresholds being set in the requirements are appropriate (via methods such as hypothesis testing as introduced in statistics [40]) and can positively impact safety (by positively influencing the safety-related properties). Since these requirements are not exposed to the system integrators (OEM), validating these requirements is the task of the AI component/system provider.

12.6 Work products

12.6.1 AI system verification report, resulting from requirements 12.3.1 to12.3.5and 12.3.7.

12.6.2 Integrated AI system, resulting from requirements 12.3.6.

12.6.3 AI system validation report, resulting from requirement 12.3.8.

13 Safety analysis of AI systems

13.1 Objectives

The objectives of this clause are:

a) to identify safety-related faults and AIerrors that can lead to the violation of AI safety requirements;

© ISO 2024 – All rights reserved

b) to identify their potential causes;

c) to support the definition of safety measures to prevent or control safety-related AI errors;

NOTE 1 These measures can include improving AI design, AI methods, dataset generation, updating AI safetyrequirements and related AI system development processes.

d) tosupport the verification of AI safety requirements, through modification or identification of new AI safety requirements on data specifications and collection, design specifications, and test specifications.

NOTE 2 The objectives, scope, and level of granularity of the safety analysis can depend on the phases of AI safety lifecyle.

NOTE 3 Safety analysis of the AI system complements the safety analysis inaccordance with ISO 26262 and ISO 21448.

NOTE 4 Safety analysis of an AI system can be performed within the safety analysis of the encompassing system, e.g. an item or a vehicle.

NOTE 5 Dependent Failure Analysis (DFA) is an important activity that follows the safety analysis of an AI system. DFA of AI systems is a new area of research, and is not covered in this document. The reader can refer to ISO 26262-9:2018, Clause 7 for guidance on DFA, which can be applicable to AI systems.

This clause aims to provide confidence that the risk of violation of the AI safety requirement at the AI system level due to AI errors is sufficiently low, i.e. within the acceptable residual risk.

13.2 Prerequisites and supporting information

The following information shall be available at the initiation of the safety analysis activity:

a) AI safety requirements, from Clause 9;

b) input space definition(refined) , from Clause 9;

c) known insufficiencies of the AI system and the corresponding subdomains of the input space, from Clause 9;

d) AI component or AI system architecture (refined), from Clause 10;

e) dataset requirements specification, from Clause 11;

f) dataset design specification, from Clause 11;

g) dataset verification report, from Clause 11;

h) dataset validation report, from Clause 11;

i) dataset safety analysis report, from Clause 11;

j) AI system verification report, from Clause 12;

k) AI system validation report, from Clause 12.

NOTE 1 The AI component or AI system architecture (refined) can be used to determine the boundaries of the safety analysis.

NOTE 2 Safety analysis can be performed at different phases of an AI safety lifecycle. Therefore, during early phases of an AI system development, availability of the prerequisites can be limited.

© ISO 2024 – All rights reserved

NOTE 3 Safety analysis can be performed at different levels of integration, e.g. AI system, AI components, or AI models, or with different focus, e.g. architectural aspects, data aspects, or combination thereof.

13.3 General requirements

13.3.1 Safety analysis techniques suitable for identifying the safety-related AI errors of the AI models in AI systemsshall be applied.

13.3.2 Safety analysis of the AI system shall identify the AI errors of the AI system and its components that have the potential to violate one or more AI safety requirements.

13.3.3 Safety analysis shall identify the safety-related faults, potential functional insufficiencies, and their potential underlying issues of the identified safety-related AI errors, if one or more AI safety requirements are violated due to the identified AI errors.

13.3.4 Safety analysis results shall be used to identify prevention or mitigation measures to address the causes of the AI errors that are potentially violating one or more AI safety requirements.

13.3.5 Safety analysis results shall be used to verify the completeness of the AI safety requirements.

13.4 Safety analysis of the AI system

13.4.1 Scope of the AI safety analysis

Safety analysis of AI systems includes a systematic identification of AI errors in an AI system and, in particular, functional insufficiencies and safety-related faults that could lead to the violation of an AI safety requirement. These AI errors can be related to:

— an AI component consisting of an AI model;

— an AI component not consisting of an AI model.

NOTE Safety-related faults and functional insufficiencies related to an AI component can be originated in that AI component or be due to the interaction of the AI component with other components within the AI system or outside of the AI system.

When AI models are in the scope of the safety analysis, safety analysis addresses the safety-related faults and functional insufficiencies of the AI models, their causes and their impact on vehicle behaviour. Safety analysis starts early during the development. Figure 13–1 shows a flowchart of a top-down safety analysis approach in an AI system as an example. In this example, safety analysis is started upon the observation of an undesired safety related behaviour at the vehicle level, however, in general, safety analysis can start from the level required determined by the team performing the analysis. In case that safety-related faults and functional insufficiencies are related to an AI component that does not contain an AI model, safety analysis can be performed following the requirements and recommendations of ISO 26262-9:2018, Clause 8 and ISO 21448. It should be noted that during the development of an AI system, other safety analysis methods, for example, bottom-up approaches can also be used to identify the faults and potential insufficiencies which might lead to AI errors.

In general, AI errors of an AI model are either due to issues in data specification and collection or issues in design and implementation or issues in requirement specification. Safety analysis to identify issues in data specification and collection and to define safety measures for prevention, or control of safety-related issues is discussed in Clause 11.4.3 (safety analysis of datasets).

Safety analysis at the design phase identifies design related issues which can contribute to AI errors violating an AI safety requirement. Safety analysis at the design phase is discussed in Clause 10. If safety analysis

© ISO 2024 – All rights reserved

identifies insufficiencies in an AI safety requirement specification, the requirement specification might need to be modified or one or more new requirements might need to be added. Requirement modification/addition follows the guidance provided for requirement modification/addition in Clause 9.

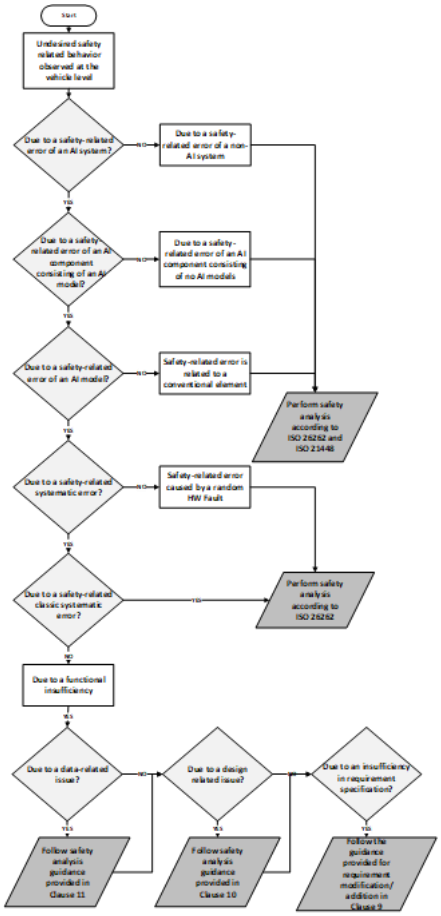


Figure 13–1 — An example flowchart for a top-down safety analysis in an AI system upon the observation of an undesired safety-related behaviour

13.4.2 Safety analysis based on the results of testing

If testing of an AI system, at any level, reveals the presence of AI errors, the results of the safety analysis are used to evaluate the impact of the AI errors on the compliance of the system under test to its AI safety requirements, to identify the causes of the safety-related AI errors and to define mitigation measures. Here, safety analysis activities consist of risk evaluation, root-cause analysis and risk mitigation as shown in the AI component testing workflow in Figure 12–3.

a) **Risk evaluation:** during this activity, the risk due to the failed test is evaluated to estimate the impact on safety. In general, if any of the AI safety requirements are violated due to an AI error, it can be concluded that safety is not achieved.

NOTE 1 A test can also fail due to the violation of non-safety requirements. In case multiple non-safety requirements are violated as the result of the failed test, the risk due the violated requirements is evaluated to assess the impact on safety.

b) **Root-cause analysis:** in this step, underlying issues for the AI error(s) are identified. Issues in an AI model, in general, can be related to many areas including:

— AI safety requirements allocated to the AI component consisting of the AI model;

© ISO 2024 – All rights reserved

— AI data including data sets and

— AI model design.

Once a potential category of causes has been identified, a more detailed safety analysis related to that area can be performed to evaluate the cause in more detail. For this, the results of safety analysis performed during AI model design or data set generation can be used. For safety analysis on data sets, see Clause 11, and for safety analysis during AI model design, see Clause 10.

c) **Risk mitigation:** When root-causes of the issues are identified, prevention, detection, and/or control measures regarding the identified root causes need to be defined. These risk mitigation measures include:

— Modification/addition/removal of AI safety requirements;

NOTE 2 For supervised machine learning, the activities in root-cause analysis and the proposal of modification/addition/removal of AI safety requirements are further detailed in Clause 9.5.3.

— Changes in the AI model;

EXAMPLE Testing of an ML model developed for detecting pedestrians in an autonomous driving system reveals that the ML model does not detect pedestrians which are standing next to a traffic post. Safety analysis shows that the hyperparameters of the neural network are not selected optimally. The hyperparameters of the AI model are changed to mitigate the issue.

— Changes in the dataset

— Modification of the AI development processes

Mitigation measures are discussed in more details in Clause 9, Clause 10, and Clause 11. These risk mitigation measures then need to be implemented as part of the AI system development including requirement derivation, design and dataset creation according to Clause 9, Clause 10 and Clause 11, respectively. This activity might require the creation of additional safety-related test cases.

13.4.3 Safety analysis techniques

Safety analysis techniques should provide adequate identification of hazards and their potential causes. The sufficiency of a safety analysis technique to model a system is argued by the following methods:

— proven-in-use-argumentation; and

— critical review of the chosen technique, where pros and cons of the technique for safety analysis of the system is evaluated and its limitations are identified.

Since safety of AI systems is a relatively new topic, the proven-in-use-argumentation is challenging to apply. Safe application of AI is challenging, because AI introduces new classes of mechanisms, and concerns how risks can emerge. The concerns include inclusion of training data instead of system specifications, no clear design as system architecture, uncertainties and the explainability challenges in the models' outputs. Some of the salient features of AI systems that can impact the safety analysis are:

— AI systems can behave nonlinearly. Depending upon their current state and context, they might react to the same inputs very differently. Additionally, smaller disturbances in the input can produce irregular outputs.

— In some cases, the environment that the AI system is deployed in is ever evolving. For example, in case of highly-automated driving vehicles operating in the open context, new traffic participants can appear over the course of their operations.

© ISO 2024 – All rights reserved

— AI systems can produce complex interactions within its elements and with the environment. The models that result for such systems might introduce complex correlations.

Safety analysis techniques analyse the systems with underlying assumptions. Some of the commonly used safety analysis methods are shown in Table 13–1. These salient features of AI systems require a thorough understanding of the safety analysis method selected to analyse these systems. Some of the existing analysis techniques have been enhanced to model the AI systems [41], [42] while newer modelling techniques have been introduced with stronger assumptions to model the AI systems [43],[44], [45].

Table 13–1 — Safety analysis techniques

Safety Analysis	Modelling Assumptions	Advantages	Limitations
Fault Tree Analysis [41]	— Independence of events — Bernoulli model — Simplified causal relation — Static temporal concept	Based on Boolean algebraic concepts	Generally static
Failure Mode and Effects Analysis [46]	— Single point of failure — Simplified causal relation — Static temporal concept —	— Documented process — Early design decision — Easy to implement	— Inability to determine complex failure mode Cause-and-effect chains (might not be predictable)
System Theoretic Process Analysis [47]	— Simplified causal relation — System fails in a certain pattern	Models' interactions Implementation is more comprehensive	Limited keyword set
Event Tree Analysis[48]	Single point of initiations	Assessment of multiple faults and failure	Probability identification is difficult
Bayesian Network/ Causal Bayesian Network [44]	— Model is the best representation — Probability distributions are known	— Can model complex relations — Hybrid modelling is possible — Models multiple point initiations and failure — Can handle conditional independence concepts	— Difficult to model — Probability identification is difficult
HAZOP [49]	Single point of initiations	— Documented process	— Single point of failure

© ISO 2024 – All rights reserved

Safety Analysis	Modelling Assumptions	Advantages	Limitations
		— Early design decision — Easy to implement	— Propagation of failures is not clear — Identification of causes is weak

13.5 Work products

13.5.1 Safety analysis report, resulting from13.3.1 to 13.3.5.

14 Measures during operation

14.1 Objectives

The objectives of this clause are:

a) to define the process requirements to continuously assureAI safety after deployment,

b) to use the measures defined in clause 8 and/or additional measures for the identification of safety risk associated with the AI system during operation and measures to maintain AI safety during operation,

c) to ensure responses are in place to address unacceptable safety risks associated with the AI system and ensure re-approval of the modified AI system before release.

14.2 Prerequisites and supporting information

The following information shall be available at the initiation of this phase:

a) AI system definition (from external sources), including:

   1) interfaces with the encompassingsystem;

   2) Assumptions on the use of the AI system;

b) field data collected by the encompassing system;

c) safety requirements on the AI system,from Clause 9;

d) AI component or AI system architecture, from Clause 10;

e) dataset requirements specification, from Clause 11;

f) dataset design specification, from Clause 11;

g) dataset maintenance plan, from Clause 11;

h) known insufficiencies of the AI system and the corresponding subdomains of the input space, from Clause 9

i) results of verification and validation activities including known functional insufficiencies of the AI system (if available),fromClause 12;

j) safetyassurance argument, from Clause Clause 8.

© ISO 2024 – All rights reserved

14.3 General requirements

14.3.1 The process and its activities necessary to assure the AI safety and the validity of the assurance argument during operation shall be specified.

NOTE 1 This process can include the procedure to terminate the safety support and notify properly to the user of AI system regarding this termination.

NOTE 2 These activities include the identification of safety issues of the AI system during operation and their resolution procedure.

14.3.2 The on-board and off-board measures necessary to execute the specified activities in 13.3.1 shall be developed and implemented.

EXAMPLE Measures can include monitor the operational status of the AI system, detect safety-related errors, etc.

14.3.3 The identified safety-related field events shall be evaluated and, if the risk is deemed unacceptable, countermeasures shall be taken to mitigate the risk.

14.3.4 The effectiveness of the countermeasures shall be evaluated after their application during the operation phase, and the countermeasures shall be modified if the residual risk is still unacceptable.

14.3.5 The specified maintenance activities during operation shall be executed in order to continuously keep AI safety to a reasonable level.

EXAMPLE Field data collection, AI re-training, re-validation and re-approval, etc. can be executed in order to continuously keep the AI safety.

14.4 Planning for operation and continuous assurance

14.4.1 Safety risk of the AI system during operation phase

Upon achieving recommendation for release, the residual risk is evaluated to be acceptable based on the evidence and assumptions generated during the development phase. However, post-deployment field risk evaluation could detect an elevated risk associated with the AI system due to hazards resulting from:

a) Development uncertainties, for example:

— Incorrect estimation of residual risk;

— Previously unknown hazardous functional insufficiencies;

— Incorrect estimation of the occurrence of AI related faults occurred during operation;

b) Incorrect unexpected operation-specific activities, for example:

— During maintenance, e.g. retrofit camera or radar without recalibration or poor tolerance;

— During update of the AI system, or update of external systems that interacted with the AI system, e.g. outdated software version, out of sync of component updates;

c) Changes in the operation environment, for example:

— New traffic rules;

— New traffic facilities;

© ISO 2024 – All rights reserved

— New type of traffic participants;

— Changes of assumptions on the operating conditions.

NOTE Changes in the operation environment could introduce OOD samples and potentially cause OOD errorsin the AI system.

Specific activities can be necessary during the operation phase to address these risks and assure a continuous level of AI safety.

14.4.2 Safety activities during the operation phase

Figure 14–1 illustrates the flow of the activities of this clause to assure the safety of AI during the operation phase.

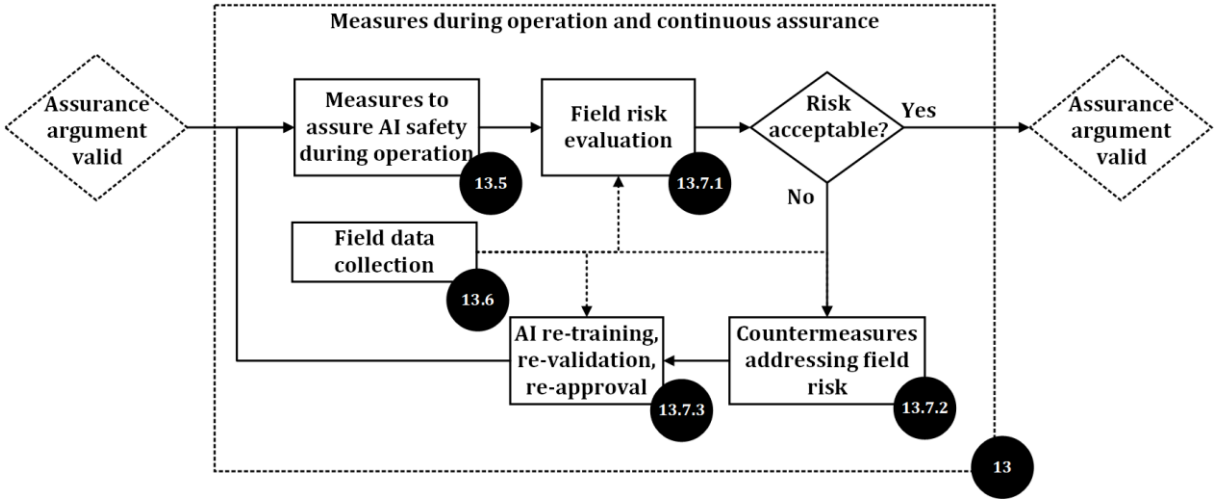


Figure 14–1 — Safety assurance during operation phase[2]

Safety assurance during operation starts with applying measures to assure AI safety in the operation phase (14.6). Technical measures are applied to monitor the behaviour of AI system during operation phase, and if anomalies or undesired safety-related behaviour at the vehicle level are detected, mitigation measures are taken (14.6.2). Some additional measures are introduced to address misuse related risk, e.g. user guidance (14.6.3). Field data will be collected during monitoring (14.7), in order to support afterwards risk evaluation, mitigation and AI system modification.

Risks identified in the field are evaluated (14.8.1). If it is shown that the level of identified risk is acceptable, then no further activities are applied. Alternatively, if the risks are found to be unacceptable, countermeasures are determined based on the evaluation results (14.8.2). The AI system can go through re-training, re-validation and re-approval process, if necessary (14.8.3).

14.5 Continual, periodic re-evaluation of the assurance argument

Due to the complexity of the functionality to be implemented, the environment in which it is deployed, as well as the nature of the AI technologies themselves, some uncertainty in the assurance argument might inevitably remain.

This leads to a residual risk that the safety requirements allocated to the AI system might be violated during operation. This residual risk can be related to previously unknown triggering conditions of residual

[2] Safety assurance activities during operation phase rely on previous clauses, i.e. 12.5 refers to Clause 10, 12.6 refers to Clause 11, 12.7 refers to Clause 9Clause 10Clause 12Clause 13Clause 8.

© ISO 2024 – All rights reserved

insufficiencies in the AI system, inadequacies of the assurance argument, or to changes within the operating context. A continual, periodic re-evaluation of the assurance argument can offset this emergent risk.

The residual risk can be offset by operational restrictions, until sufficient evidence can be collected to increase confidence in the assurance argument. Examples of operational restrictions that can be applied include:

— restricting the set of operational conditions, thus reducing the risk of violating assumptions in the assurance argument;

— limiting the functionality of the AI system, thus reducing the severity of residual errors.

The re-evaluation of the assurance argument can be performed based on the criteria outlined in 8.7. In particular, evidence collected during operation can be re-used to provide additional support for claims of the assurance argument, as well as to identify potential defeaters to these claims.

EXAMPLE An assurance argument uses a set of assumptions on the input space to define the context of the assurance argument. A run-time anomaly detection identifies out-of-distribution inputs that were not considered within the set of assumptions.

In reaction to this new information, the assurance argument is re-evaluated using a wider set of assumptions, which includes the identified out-of-distribution inputs.

Triggers for the re-evaluation of the assurance argument can include:

— periodic review;

— collection of observations that can be used as additional evidence in the assurance argument;

— analysis of reported field incidents;

— results of on-board and off-board monitoring;

— change in operational parameters or environment conditions;

— modification or maintenance of the encompassing system and

— changes in operating procedures.

14.6 Measures to assure safety of the AI system during operation

14.6.1 General

The intention of this section is to give guidance for applying measures to assure AI safety during operation and trigger potential updates to the AI system. This section also provides additional non-technical measures, for example user involvement to prevent the misuse and to assure the safe operation of AI, if possible.

14.6.2 Technical safety measures

Monitoring, detection, and mitigation, which are used to evaluate the behaviour of AI systems against the errors or insufficiencies, are measures used to assure safety of the AI system during operation. These measures can rely on the on-board mechanisms and/or off-board mechanisms (e.g. cloud monitoring).

NOTE 1 The architectural measures to assure safety of AI during the operation phase are defined in Clause 10

NOTE 2 In contrast with the on-board mechanisms which are used to detect and mitigate abnormal behaviours of AI system and the vehicle equipped with AI, the off-board mechanisms can detect abnormal behaviours with higher accuracy due to, for example, larger computing power and a more precise model.

© ISO 2024 – All rights reserved

NOTE 3 The off-board measures (e.g. cloud monitoring) can also be used to monitor the general behaviours of all vehicles equipped with the AI systems. These off-board measures can then be used to support the evaluation of the overall risk after deployment of the AI systems.

Regarding monitoring and detection, the following events, including related context can be reported with the purpose of finding insufficiencies or errors of AI system, if applicable:

a) Input space related events:

— Detection of out-of-distribution-data and data distributional shift;

— Detection of exiting from operating context;

— Detection of conceptdrift or changes in features (e.g. new objects, different behaviours, new or changed rules, etc.) ;

b) Model behaviour related events:

— Detection of abnormal behaviour;

NOTE 4 Some abnormal behaviours could be caused by rare input conditions and these behaviours could be evaluated as safe after detection. For example, while reversing, the AI model stopped at a shorter distance than specified from a parked car. Behaviour is detected by the system and logged. After analysis of the report, the behaviour is considered safe and an update is not necessary.

c) Output related events:

— Detection of abnormal output;

NOTE 5 Exercise caution when implementing plausibility checks as these can lead to missed objects and safety concerns (e.g. rejecting humans taller than 7 ft could lead to mis-detection of a pedestrian carrying a flagpole or on stilts or having a child on their shoulders, etc.)

— Detection of output bias;

— Outputs with low confidence level;

d) Incidents/accidents analysis:

— Incidents/accidents where the AI system was directly or indirectly involved are analysed to support the improvement of the AI system.

NOTE 6 For detected errors or insufficiencies of the AI system which can lead to a hazard, the risk can be mitigated by measures within the AI system or the encompassing system. For example, switching to non-AI system or executing a manoeuvre that results in a minimal risk condition.

As the insufficiencies of the AI system can influence the behaviour of the encompassing vehicle system, any abnormal behaviours or emergency events of the vehicle might also imply or influence on insufficiencies of the AI system, for example:

— function degradation;

— take-over request;

— emergency manoeuvre;

— transition to a minimal risk condition;

© ISO 2024 – All rights reserved

— collision or near-collision event;

— contradiction between AI system and non-AI system

EXAMPLE The AI system and non-AI systems, which are both used for decision making, mayprovide diametrically opposed results under an unprotected left-turn scenario, for instance, one for “yield”, and the other for “not yield”.

Besides triggering modification activities, errors or insufficiencies of the AI system identified during operation may indicate the weaknessesin the development and safety assurance process, architectural measures or incorrectness of their usage assumptions, thus modification of these measures may be needed.

14.6.3 Safe operation guidance and misuse prevention in the field

The user of the AI system can lack understanding of its capabilities which results in misuse due to overconfidence in the AI system. As a potential prevention for overconfidence, users are made aware of the limitations of AI systems via, for example, user training if possible or relevant information through the human-machine interface.

EXAMPLE The user is trained to correctly use the AI system and be informed of scenarios in which the AI system is intended for use, considering the performance limitations of the AI system within these scenarios.

Another possible prevention of misuse are technical measures (e.g., warning or, degradation or disablement of services) that are triggered when the AI system is misused by the userduring operation.

14.7 Field data collection

The intention of this section is to introduce field data collection as a supplementary data source for AI system maintenance to improve dataset integrity, distribution and usage (see Clause 11)

NOTE 1 Field data collection is related to AI systems whose safety can be affected by field conditions. An autonomous driving system that makes use of AI technologies is a typical case and selected as example within this section.

The motivation to collect field data during operation includes, for example, addressing environment changes which may affect the behaviour of AI system, identifying and removing residual insufficiencies and collecting additional training data. The quality of the collected field data needs to be ensured and the data needs to be transmitted to relevant parties (e.g. manufacturers, suppliers and/or regulators) for use to support the update of the AI system if necessary. The following topics can be considered when collecting field data:

a) Competence management

To ensure the efficiency and quality of the field data collection, competence management measures can be applied to persons responsible for field monitoring, data collection or data analysis. All systems involved in field monitoring activities will be tested, validated and released to ensure required reliability level.

b) Data characteristics

The data characteristics of the field data can be defined dependent on the planned usage of the data.

EXAMPLE 1 In some cases, a large number of images of a high resolution are needed in order to improve the 2D image perception performance of the AI system, such as classifying a certain type of traffic sign. In other cases, the AI-based image processing algorithm might be dependent upon relationships between sequences of images over time. For such cases, a minimum length of video sequences along with other associated sensor data and the results of the current iteration of the AI system are required to improve performance.

When analysing the field data, the following data characteristics can be considered:

NOTE 2 The data characteristics given below are not exhaustive.

— data categories,

© ISO 2024 – All rights reserved

EXAMPLE 2 Data source (radar, LiDAR, camera or HD map).

— data content,

EXAMPLE 3 Vehicle Identification Number (VIN), images from front camera, parsing data from front perception, changes of control mode, received remote control command, operation status and HMI data.

— data format,

EXAMPLE 4 JPEG, PNG and BIN.

— data size.

c) Data collection trigger and transfer

To ensure that the collected data is sufficient to identify, analyse and improve safety-related issues, clear data collection triggering criteria are defined including the triggering conditions, triggering interval, start time and end time, triggering priority according to different cases.

EXAMPLE 5 The triggering rules of field data collection can be:

— accident or incident: collision event involving the automatic driving vehicle equipped with AI system;

— functional termination: autonomous function failure/insufficiencies, terminated by the human taking over;

— exiting operational design domain (ODD): the specific objects are detected, such as the red light, stop signs, etc. which are not within the ODD scope for a highway pilot feature, the value reported by rain sensor exceeds the threshold for a feature designed for no or small rain weather;

— implausible events: the distance or speed jitter of the detected object exceeds a certain value, the distance deviation of the detected object is greater than a certain value measured by different sensors;

— other functional insufficiencies: the target motion predicted by the AI algorithm is inconsistent with the actual situation;

— diverging decisions of redundant diverse AI elements or between AI and non-AI based elements.

NOTE 3 Triggering rules can be updated over-the-air (OTA) in order to collect different kinds of data. To collect sufficient data for each event, a timing buffer can be considered, for example: recording starts from at least X s before the event to at least Y s after the event.

When transferring the data, the conditions that may affect the reliability of the transfer are considered, for example: the data transfer is interrupted by loss of power and may therefore lead to loss of data.

d) Data storage

To ensure the integrity of data storage, safety mechanisms are implemented where reasonably practicable, for example, adding data integrity protection. The operation conditions that may influence the data storage are also considered.

NOTE 4 The general data storage requirements used for AI data collection can also be used for field data storage.

e) Configuration information

To ensure correctness of data collection, the configuration information about the field data to be collected is specified, which may include access rights, tools and repositories, and aligned with the requirements for datasets (see 11.3)

© ISO 2024 – All rights reserved

14.8 Evaluation and continuous development

14.8.1 Field risk evaluation

Based on field measures (14.6) and data collected (14.7), the accidents, anomalies and undesired safety-related behaviour at the vehicle level potentially related to AI systems can be manually or automatically reported to the manufacturers or service providers. The number of reported issues might be large during the early phase after deployment. In order to solve the reported issues efficiently and economically, the manufacturers or service providers investigate the causes and evaluate the field risk of the issues, to determine the proper reactive actions to be taken, such as recall or OTA update.

The field risk evaluation is different compared to the hazard risk evaluation during the development phase. In particular field risk evaluation is based on the real consequence of issues occurred during the operation instead of assumptions or estimations made at development phase.

To objectively evaluate the effects of the issues, the probability of occurrence, the severity and the effectiveness of countermeasures addressing the risk of the existing issues can be considered. This is similar to the occurrence, severity and detection parameters used by FMEA method for a systematic evaluation of risk.

NOTE 1 Alternative risk evaluation methods to FMEA based on a systematic methodology and predefined criteria can also be applied.

a) evaluation of the probability of occurrence

As described in Figure 6–12, safety-related issues of the AI system can be caused by random hardware faults and/or systematic factors (e.g. systematic faults or functional insufficiencies).

— For issues associated with random hardware faults, the occurrence considered is determined by the failure rate and the probability of exposure to a hazardous scenario which has been considered by ISO 26262;

— For issues associated with systematic faults or functional insufficiencies, the risk is mainly determined by the probability of the exposure to the critical situations or probability of triggering events;

— As the quantity and location of the vehicles can be known at this phase, it is possible to provide the occurrence with higher accuracy than during development.

EXAMPLE The occurrencerate of the issue over a given time period can be predicted based on the failure rate of the component, the quantity of vehicles in the field and the probability of the vehicles facing the hazardous scenarios.

b) evaluation of the severity

The severity evaluation can be based on the method defined by ISO 26262-3, which recommends Abbreviated Injury Scale (AIS) ranking method.

NOTE 2 In addition, other issues that can cause loss due to cybersecurity risk, violation of traffic rules or serious customer complaints might also be considered.

c) evaluation of the detection and mitigation measures

The measures in 14.6 can help to detect and mitigate the risk of errors in the AI system. The potential controllability by the driver can also be considered as a mitigation of the issue, if the field data shows relevant evidence.

It is possible to give risk evaluations based on the factors above in a qualitative way or quantitative way (if rates are defined for each factor). The evaluation results will support theidentification of the response actions to be taken.

© ISO 2024 – All rights reserved

NOTE 3 For serious accidents (e.g. fatalities), even if the occurrence had been rated as low or detection as high based on predefined criteria, the rating criteria can be adjusted and risk can be considered differently.

14.8.2 Countermeasures addressing field risk

The safety development of AI systems does not end after the safety release. The field risks could be higher than expected in case the on-board measures cannot detect and mitigate all risks. If hazardous events occur in the field, the following additional countermeasures can be taken:

— Issue investigation actions to determine the causes of risk, e.g. scenario reconstruction based on the data collected, especially for AI-related incidents or accidents;

— Risk evaluation as introduced in 14.8.1;

— Restrictions on context of use or functionality deactivation or replacement;

— Update of the AI system, for example over-the-air (OTA) update, when unacceptable systematic faults or insufficiencies are identified

— Customer notification, which can be taken together with the restrictions on context of use or AIsystem update actions, or dedicated notification to address misuse risk, e.g. emphasizing the operation requirements to the passengers by placing a warning card in the robotaxi.

NOTE Depending on the urgency of identified field risks, immediate actions or long-term actions could be taken based on risk evaluation.

An appropriate issue management process is important to ensure the effectiveness of countermeasures, including incidents or accident reporting, issue investigation, risk evaluation and countermeasure management processes.

The effectiveness of the countermeasures taken are monitored and evaluated after implementation and adjusted, if the risk is still unreasonable.

14.8.3 AI re-training, re-validation, re-approval and re-deployment

The system can be incrementally developed onthe basis of collected field data and countermeasures to compensate for the identified risks, making the system safer and more robust. The intention of this section is to introduce AI model re-training, re-validation, re-approval and re-deployment.

NOTE AI system update and re-approval involves the activities described in Clause 7toClause 13, if relevant.

a) re-training

During operation, valuable field data can be collected. Together with the data from the previously trained model, this data can be used to re-train the new model with the expectation of better performance. Re-training can be achieved by fine-tuning the pre-trained model or by training from scratch.

— Fine-tune: fine-tuning refers to small adjustments to model parameters. The newly acquired field data can be used to fine-tune the released model. When fine-tuning, a small learning rate is used so as not to over-distort the existing model.

EXAMPLE 1 For some DNN specific multi-task networks, it is often the case that only one specific task head needs to be fine-tuned while freezing the backbone and other task heads, for example, only the detection head can be fine-tuned when input training data are labelled for detection.

— Train from scratch: usually after a long period of operation, all parameters of the model can be randomly initialized to re-train the model from scratch. This approach is expected to get better performance than fine-tuning in the original model. However, compared to fine-tuning, this method requires more data

© ISO 2024 – All rights reserved

volume, computing time, and computing resources. Using a pre-trained backbone is a common method. Applying an existing backbone to train on the desired task can reduce the computational cost and speed up the convergence.

b) re-validation

After re-training, the updated AI model is integrated into AI system, and re-validated to provide evidence that the safety-related issues are solved and all relevant safety requirements are met.

EXAMPLE 2 The datasets of known issues can be used to re-validate the updated AI system in virtual or real-world testing, or a combination of both, and to demonstrate the absence of safety performance degradation, if applicable.

c) re-approval and re-deployment

After an update to the AI system , the safety assurance argument is re-evaluated (see clause 11). Once re-approved, the AI system update can be deployed.

14.9 Work products

14.9.1 The specification of the process and its activities for assuring AI safety during operation, resulting from 14.3.1.

14.9.2 The specification of the necessary off-board and on-board measures, resulting from14.3.2.

14.9.3 Field data and functional insufficiencies detected during operation, resulting from 14.3.3.

14.9.4 Evidence of the effectiveness of measures for ensuring AI system during operation, resulting from 14.3.4.

14.9.5 Evaluation report of functional insufficiencies detected during operation, and **updated version of the safety assurance argument** if applicable, resulting from 14.3.3 and 14.3.5.

15 Confidence in use of AI development frameworks and software tools used for AI model development

15.1 Objectives

The objective of this clause is:

a) to provide requirements and guidance to identify, mitigate and document possible sources of errors and inappropriate biases in the off-line processes, tools and principles used to develop, verify and deploy safety-related AI models.

15.2 Prerequisites and supporting information

The following information shall be available at the initiation of this activity:

a) documentation of development processes and tools used within the AI safety lifecycle (from external sources);

b) AI system-specific development measures and procedures (from Clause 7to Clause 14).

15.3 General Requirements

15.3.1 Processes,tools, and work products used for the development of safety-related AI models shall be analysed to identify, mitigate and document possible sources of errors.

© ISO 2024 – All rights reserved

EXAMPLE Errors can be caused by inappropriate biases in processes such as field data collection, labelling, sampling, tools such as data processing, deep learning frameworks, work products such as data, AI models.

NOTE The approaches discussed in ISO/IEC TR 5469:2024 11.5.3 and ISO 21448:2022 D.2.5 canbe used to analyse offline training processes.

15.3.2 Confidence shall be demonstrated that software tools used to develop, verify and deploy safety-related AI models are suitable to be used to support activities or tasks required by this document.

NOTE ISO 26262-8:2018 Clause 11 canbe used to demonstrate confidence in the use of software tools.

15.3.3 Appropriate principles for data-driven AI models shall be applied to training and evaluation to ensure control or avoidance of safety-related faults in the AI models.

NOTE A specific level of robustness and quality in software, including data-driven AI models, is essential for assuring safety. Design principles that govern software unit design and implementation at the source code level, such as enforcing a single entry and exit point in subprograms and functions, have traditionally been employed to attain the desired quality and robustness in conventional software. However, these principles are applicable solely to the software implementation aspect of data-driven AI models that have been trained and evaluated on data. Therefore, appropriate principles are necessary for ensuring the requisite level of robustness and quality in the training and evaluation aspects of data-driven AI models.

EXAMPLE The influencing factor classes listed in Table 9–3and their managing approaches elaborated in Table 9–1 can be used as principles.

15.4 Confidence in the use of AI development frameworks

A robust process used to develop AI models will reduce the risk of introducing errors in the development of the AI system, thereby making the AI system safer. Specific analysis depends on each system, e.g., automated emergency braking systems and driver status monitoring systems. Typically, AI models are developed using a multi-step process such as the onegiven in Figure 15–1. The offline training used in the process can be a source of errors in the final AI model. ISO/IEC TR 5469:2024 11.5.3 PFMEA of offline training of AI technology proposes the use of a Process Failure Mode and Effects Analysis (PFMEA) to analyse AI offline training. ISO 21448:2022 D.2.5 Analysis of the off-line training process of machine learning algorithms describes a similar analysis approach used for SOTIF issues.

© ISO 2024 – All rights reserved

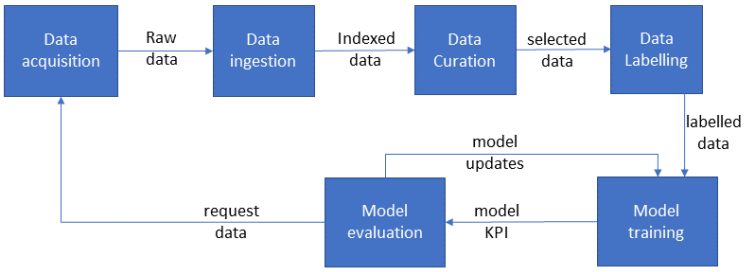


Figure 15–1 — Example offline multistep AI training process for PFMEA

PFMEA is a well-known technique in the automotive industry [50]. PFMEA is an inductive method often applied to manufacturing processes. The analogy is that the offline training process is "manufacturing" an AI model and many of the benefits of a PFMEA apply. It is beyond current technology to trace AI's safety-related systematic issues to root causes, e.g., training and deployment errors, SOTIF issues, etc. Therefore, the overall integrity of training processes, which can be a source of errors in the final AI model, is analysed during AI development. The perspectives used in the safety analysis of systems, e.g., four influence factor classes described in Table 9–1, are connected to PFMEA.

A PFMEA finds failure modes in each element of the AI training processes. Then, their effects on subsequent processes and countermeasures to detect such failures are reviewed. Table 15–1describes examples of potential failure modes and effects in the AI training processes of Figure 15–1which could result in a safety-related systematic issue, i.e., performance insufficiencies and safety-related classic systematic faults as included in Figure 6–12.

Table 15–1 — Example of potential failure modes and effects in PFMEA

Process	Potential failure modes	Potential effects
Data acquisition Description: Process step for collecting data to be used in model training and test.	Specific scenes are missing intest datasets.	The model has degraded performance in scenarios involving missing scenes.
	Test data coverage is biased.	In the model evaluation process, evaluation results are biased.
	Only a small number of routes are planned for data collection, and the collected training and test datasets lack variation.	Due to lack of variation, the model training process results in low-performance models, and test results are unreliable in the model evaluation process.
	Unintended data collection scenarios are used, and the collected datasets haveinappropriateattributes (meta	The mixture of training data samples using data attributes in the model training process and scene-wise evaluation based on data attributes

© ISO 2024 – All rights reserved

	labels), e.g., weather and time of the day.	in the model evaluation do not work as intended.
Data ingestion Description: Process step for uploading collected data to servers used for off-line ML model training	Data is corrupted during upload from data collection vehicle to cloud storage	Corrupted or lost data during training
Data curation Description: Process step generates input data set for further labelling and to be used for training	Curation recognises edge cases as outliers and unintentionally excludes them.	In the model training process, the trained models have performance degradation for these edge cases.
Data labelling Description: Process step identifies and labels objects within the data set to be used for model training	Objects carried or pushed by pedestrians may be included or excluded in the bounding box, leading to inconsistent labelling.	In the model training process, the trained models perform differently for different labelled objects.
	Labeller has bias (e.g. omits to label motorcycles)	The training models have performance degradation due to bias.
	Displayed labels and recorded labels are different in data labelling tools.	In the model training process, the trained models learn the wrong labels.
Model training Description: Process step to create trained model from labelled data (e.g. Figure 11–2)	Static random seeds are used during the development for debugging, and these are left in the production code.	In the model training process, random number generators do not work appropriately, and machine learning and hyperparameter optimisation frameworks do not work as intended. As a result, the trained models have consistently low performance.
	Unintended training and test data sets are loaded.	Within the model training process, the trained model is optimised to different contexts, and the trained models have consistently low performance.
Model evaluation Description: Theprocess step verifies whetherthe model meets KPIs. A decision is then made to continue with more training, collect more data or end training	The harmonic mean of precision and recall is specified as an evaluation metric, but only recall was evaluated.	As a result, the trained models become recall oriented, i.e., many false positives and few false negatives, which does not meet system requirements.
	Training datasetsare leaked to AI test datasets. AI testdatasets are not covering the ODD in a suitable manner.	In the model evaluation process, the evaluation results are not reliable or are overestimated.

Each step of the process can be further broken down (e.g. model training broken down to flow of Figure 11–2) for a more detailed analysis.

The process analysis may begin as soon as one has a basic understanding of the considered process's inputs, outputs, and internal architecture, even if that means proceeding without a complete requirements specification or a complete architecture specification of the process. This iterative process analysis may lead to the specification of additional requirements and process updates. The analysis completes by considering the fully refined architecture and requirements. Even though the analysis focuses on the process for the

© ISO 2024 – All rights reserved

creation of the AI model, the process analysis can be iterative and may also influence architectural and design decisions.

Example information to start a PFMEA:

— Process flow diagram(s) showing the entire AI model creation process flow

— Block diagram of individual process steps including internal process steps showing major components of the process step

— Boundary showing what is the scope of the analysis, the neighbouring process steps to the considered process element

— Identified purpose(s) of the individual steps

— Conceptual data flow(s) between the considered process step, its neighbouring steps, and its internal components

— List of tools used in the AI model creation process

— Training and evaluation requirements

15.5 Confidence in the use of tools used to support the AI-safety lifecycle

The training of AI models often involves tools (e.g. data labelling tools, machine learning frameworks, and hyperparameter optimisation frameworks) to train or optimize models. Tools may also be used in the labelling and curation of data along with other steps in the training process. These tools are potential sources of training and deployment errors. ISO 26262-8:2018 Clause 11 can be used to ensure that the tools do not cause an unreasonable safety risk.

15.6 Principles for data-driven AI model training and evaluation

Training and (data-driven) evaluation form key parts of the development of data-driven AI models. The causes of insufficiencies of data-driven AI models are classified into the influencing factor classes (Table 9–1), as depicted in Figure 9–3. The certainty in influencing factor classes during the AI model development process ensures the development quality of data-driven AI models. Uncertainties in influencing factor classes can impact a multi-step process such as given in Figure 15–2. Table 9–1elaborates on the approaches to manage the certainty of influencing factor classes, and these can be used as the principles for data-driven AI model training and evaluation.

© ISO 2024 – All rights reserved

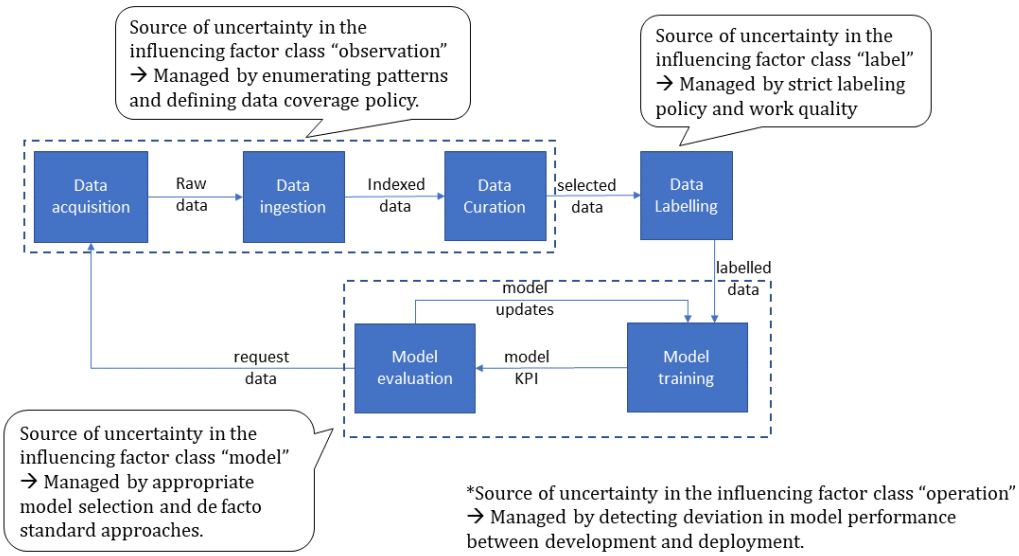


Figure 15–2 — Example offline multistep AI training process and influencing factor classes

15.7 Work products

15.7.1 Evidence for the analysis of the AI model creation processes,resultingfrom 15.3.1.

15.7.2 Evidence for the confidence in the software tools,resulting from 15.3.2.

15.7.3 Evidence for the execution of the AI model creation processes with the principles,resulting from 15.3.3.

© ISO 2024 – All rights reserved

Annex A

Overview and workflow of ISO PAS 8800

Table A–1 — Summary of the normative clauses of this document

Clause	Objectives	Pre-requisites	Work products
Clause 7) **AI safety management**	a) to define an AI safety lifecycle and its activities to ensure that contributing errors of the AI system do not lead to unreasonable risk of undesired safety-related behaviour at the vehicle-level; b) to ensure that overall and project specific safety management processes and activities are appropriate to ensure the safety of the AI system; c) to plan, initiate and conduct the AI safety activities.	a) the AI system definition (from external sources), including: 1) the AI system functionality; 2) the interfaces of the AI system with the encompassing system, including if applicable, the ASIL capability of the inputs to the AI system. 3) the safety requirements allocated to the AI system, including if applicable: i) the ASIL value of the safety requirements; ii) the acceptance criteria or validation targets derived in compliance with ISO 21448:2022, Clause 6 or 9.	7.6.1 **AI safety lifecycle** resulting from 7.3.1 to 7.3.4. 7.6.2 **Work products of ISO 26262-2:2018, 5.5**, resulting from 7.3.4. 7.6.3 **Work products of ISO 26262-2:2018, 6.5**, resulting from 7.3.3 and 7.3.4, in particular the safety plan. 7.6.4**Work products of ISO 26262-2:2018, 7.5**, resulting from 7.3.4.
Clause 8) **Assurance arguments for AI systems**	a) to develop an assurance argument demonstrating that the safety requirements allocated to the AI system are fulfilled; b) to evaluate whether the assurance argument reflects the actual residual risk of the AI system violating its safety requirements;	a) the AI system definition (from external sources), including: 1) a specification of the safety requirements allocated to the AI system; 2) a definition of the technical context within the encompassing system (e.g. definition of interfaces, conditions under which the AI system functionality is triggered, etc.); 3) a specification of the input space; b) requirements on the assurance argument and work products for the AI system (from external sources). These requirements can be derived from the	8.8.1**Safety assurance argument**, resulting from 8.3.1, 8.3.2. 8.8.2**Confirmation measure reports**, resulting from 8.3.3.

© ISO 2024 – All rights reserved

Clause	Objectives	Pre-requisites	Work products
		assurance argument of the encompassing system as well as safety management procedures from Clause 7; The following information shall be available for the finalization of these activities: c) the work products of the AI safety life cycle;	
Clause 9) **Derivation of safety requirements on AI systems**	a) to specify a complete and consistent set of safety requirements on the AI system, that are sufficient to ensure AI safety; b) to refine AI safety requirements based on learnings from development, verification and validation; c)to specify the limitations of an AI system over its input space to be escalated to its encompassing system development process.	a) AI system definition (from external sources), including: 1) safety requirements allocated to the AI system; 2) input space definition; 3) functional requirements; 4) impacted stakeholders; 5) the interfaces of the AI system with the encompassing system, including if applicable, the ASIL capability of the inputs to the AI system; 6) interfaces to the environment, if applicable.	9.6.1**Input space definition (refined)**, resulting from 9.3.1 and . 9.6.2**AI safety requirements**, resulting from9.3.2, 9.3.3, 9.3.4, and 9.3.6. 9.6.3**Known insufficiencies of the AI system and the corresponding subdomains of the input space**, resulting from9.3.5.
Clause 10) **Selection of AI Technologies, AI Measures and design-related considerations**	a) to select and justify appropriate AI technologies for use in the AI system; b) to identify appropriate architectural and development measures to fulfil the safety requirements prior to deployment; c)to identify appropriate architectural measures to mitigate residual functional insufficiencies of the AI system revealed after deployment; d)to identify measures for ensuring the safety requirements of the AI system are fulfilled within its target execution environment.	a) safety requirements on the AI system, from Clause 9: b) training and validation datasets, from Clause 11; c)AI component or AI system architecture, if already existing; d) AI component or AI system development process, if already existing.	10.6.1**AI component or AI system architecture** (refined), resulting from 10.3.1 to 10.3.11. 10.6.2**AI component or AI system development process** (refined), resulting from 10.3.1 to 10.3.11. 10.6.3**Implemented AI component,** resulting from 10.3.12.
Clause 11) **Data-related considerations**	a)to define the dataset lifecycle of activities related to the gathering, creation, analysis,	a)AI system definition, including:	11.5.1**Dataset life cycle**, resulting from 11.4.2

© ISO 2024 – All rights reserved

Clause	Objectives	Pre-requisites	Work products
	verification and validation, management, and maintenance of the datasets used in the development of the AI system; b)to identify the dataset insufficiencies that may impact the safety of the AI system; c) to identify the data-related safety properties that have a bearing on the safety of the AI system and that support dataset safety analysis; d)to define the countermeasures to prevent or mitigate dataset insufficiencies using dataset safety analysis methods at different steps in the dataset lifecycle; e)to define the data-related work products that support providing evidence of the safety of the AI system.	1) AI safety requirements, from Clause 9; 2) input space definition(refined), from Clause 9; b)field data and functional insufficiencies detected during operation, from Clause 14; c)safety analysis report, from Clause 13.	11.5.2**Evidence for the outputs of the defined phases of the dataset life cycle,** resulting from **11.3.3**. 11.5.3**Evidence for the safety analyses of the dataset,** resulting from 11.3.4 and 11.3.5. 11.5.4**Dataset requirements specification,** resulting from 11.3.6 and 11.3.7.
Clause 12) **Verification and validation of AI systems**	a) to verify that the AI system fulfils its AI safety requirements; b) to validate that the safety requirements allocated to the AI system are achieved when integrating into the encompassing system;	a) Safety requirements allocated to the AI system (from external sources); b) AI safety requirements, from Clause 9; c) Known insufficiencies of the AI system and the corresponding subdomains of the input space, from Clause 9; d) Input space definition (refined), from Clause 9; e) AI Component or AI System Architecture, from Clause 10; f) Implemented AI component, from Clause 10; g) Dataset lifecycle, from Clause 11; h) Evidence for the outputs of the defined phases of the dataset lifecycle, from Clause 11;	12.6.1**AI system verification report**, resulting from requirements 12.3.1 to 12.3.5 and 12.3.7. **12.6.2 Integrated AI system** , resulting from requirements 12.3.6. 12.6.3**AI system validation report**, resulting from requirement 12.3.8.

© ISO 2024 – All rights reserved

Clause	Objectives	Pre-requisites	Work products
		i) Evidence for the safety analyses of the dataset, from Clause 11; j) Dataset requirements specification, from Clause 11.	
Clause 13) **Safety analysis of AI systems**	a)to identify safety-related faults and AI errors that can lead to the violation of AI safety requirements; b)to identify their potential causes; c)to support the definition of safety measures to prevent or control safety-related AI errors; d)to support the verification of AI safety requirements, through modification or identification of new AI safety requirements on data specifications and collection, design specifications, and test specifications.	a) AI safety requirements, from Clause 9; b) input space definition(refined) , from Clause 9; c) known insufficiencies of the AI system and the corresponding subdomains of the input space, from Clause 9; d) AI component or AI system architecture (refined), from Clause 10; e) dataset requirements specification, from Clause 11; f) dataset design specification, from Clause 11; g) dataset verification report, from Clause 11; h) dataset validation report, from Clause 11; i) dataset safety analysis report, from Clause 11; j) AI system verification report, from Clause 12; k) AI system validation report, from Clause 12.	13.5.1**Safety analysis report**, resulting from 13.3.1 to 13.3.5.
Clause 14) **Measures during operation**	a)to define the process requirements to continuously assure AI safety after deployment, b)to use the measures defined in clause 8 and/or additional measures for the identification of safety risk associated with the AI system during operation and measures to maintain AI safety during operation, c) to ensure responses are in place to address unacceptable safety risks associated with the AI system and ensure	a) AI system definition (from external sources), including: 1) interfaces with the encompassing system; 2) Assumptions of the use of the AI system; b) field data collected by the encompassing system; c) safety requirements on the AI system, from Clause 9; d) AI Component or AI System Architecture, from Clause 10;	14.9.1**The specification of the process and its activities for assuring AI safety during operation**, resulting from 14.3.1. 14.9.2**The specification of the necessary off-board and on-board measures**, resulting from14.3.2. 14.9.3**Field data and functional insufficiencies detected during operation**, resulting from 14.3.3. 14.9.4**Evidence of the effectiveness of measures for ensuring AI**

© ISO 2024 – All rights reserved

Clause	Objectives	Pre-requisites	Work products
	reapproval of the modified AI system before release.	e)dataset requirements specification, from Clause 11; f) dataset design specification and maintenance plan, from Clause 11; g) dataset maintenance plan, from Clause 11; h) known insufficiencies of the AI system and the corresponding subdomains of the input space, from Clause 9 i) results of verification and validation activities including known functional insufficiencies of the AI system (if available), from Clause 12; j) safety assurance argument, from Clause Clause 8.	**system during operation**, resulting from 14.3.4. 14.9.5**Evaluation report of functional insufficiencies detected during operation**, and **updated version of the safety assurance argument** if applicable, resulting from 14.3.3 and14.3.5.
Clause 15) **Confidence in use of AI development frameworks and software tools used for AI model development**	a)to provide requirements and guidance to identify, mitigate and document possible sources of errors and inappropriate biases in the off-line processes, tools and principles used to develop, verify and deploy safety-related AI models.	a)documentation of development processes and tools used within the AI safety lifecycle (from external sources); b) AI system-specific development measures and procedures (from Clause 7 to Clause 14).	15.7.1**Evidence for the analysis of the AI model creation processes**, resulting from 15.3.1. 15.7.2**Evidence for the confidence in the software tools**, resulting from 15.3.2. 15.7.3**Evidence for the execution of the AI model creation processes with the principles**, resulting from 15.3.3.

© ISO 2024 – All rights reserved

Annex B

Example assurance argument structure for an AI-based vehicle function

B.1 General

This Annex is informative and provides an example of how an assurance argument for the safety of an AI system based on the principles outlined in this document can be expressed using the goal structuring notation (GSN)[6]. The assurance argument structure is expressed as an argument pattern that is intended to be instantiated for a given AI system.

The assurance argument depicted within this Annex is for illustrative purposes only and can be used as a starting point for AI system-specific assurance arguments. The argument is not necessarily complete and additional arguments and evidence may be required dependent on the AI system context and specific requirements.

Evidence can be referenced multiple times within the assurance argument.

Work products as defined within this document can contain multiple pieces of evidence as referenced in the assurance argument.

A description of the notation used can be found within the Goal Structuring Notation Community Standard V3 [6].

B.2 Assurance argument pattern for supervised machine learning

The assurance argument pattern described here can be used to construct an assurance argument for an AI system that makes use of supervised machine learning algorithms (e.g. DNNs). Example applications to which this pattern could apply include the use of AI for image processing tasks such as classification or object detection or predictive maintenance of safety-critical components.

The top level of the assurance argument is depicted in Figure B.2–1. Information that is to be replaced for an AI system-specific instantiation of the pattern, are indicated using the following notation: {Instantiable element}. The goal of the argument (G1) is to demonstrate that the AI system satisfies the requirements allocated to it within the overall system context. This context is defined in terms of:

— a set of assumptions on the input space (A1.1);

— a set of assumptions on the system context (A1.2);

— a definition of the functionality to be implemented by the AI system (C1.1);

— a definition of the safety requirements allocated to the AI system (C1.2).

The assurance argument also assumes that:

— quality management principles have been applied during the development of the AI system and its assurance argument (A1.3), that reduce the risk of systematic errors and increase the confidence in the assurance structure and evidence. The assurance argument is supported by a documented and repeatable development process.

— malfunctioning behaviour caused by random hardware faults or systematic faults are adequately addressed and confirmed through an additional argumentation, not described here.For example by following the guidance of the ISO 26262 series (A1.4).

© ISO 2024 – All rights reserved

— Development frameworks and tooling for the AI system do not impact AI safety (addressed by Assumption A1.5, see Clause 15), which may be justified by the use of pre-qualified development frameworks and tools.

The argument is structured to demonstrate that all potential functional insufficiencies in the AI system have been prevented, minimised, or mitigated during the specification, design and operation of the AI system (S1). This strategy makes use of a set of causes of functional insufficiencies for the type of application and applied AI technology (C1.3). These causes can include those described within this document as well as AI system-specific causes that are identified based on safety analyses (see Clause 13).

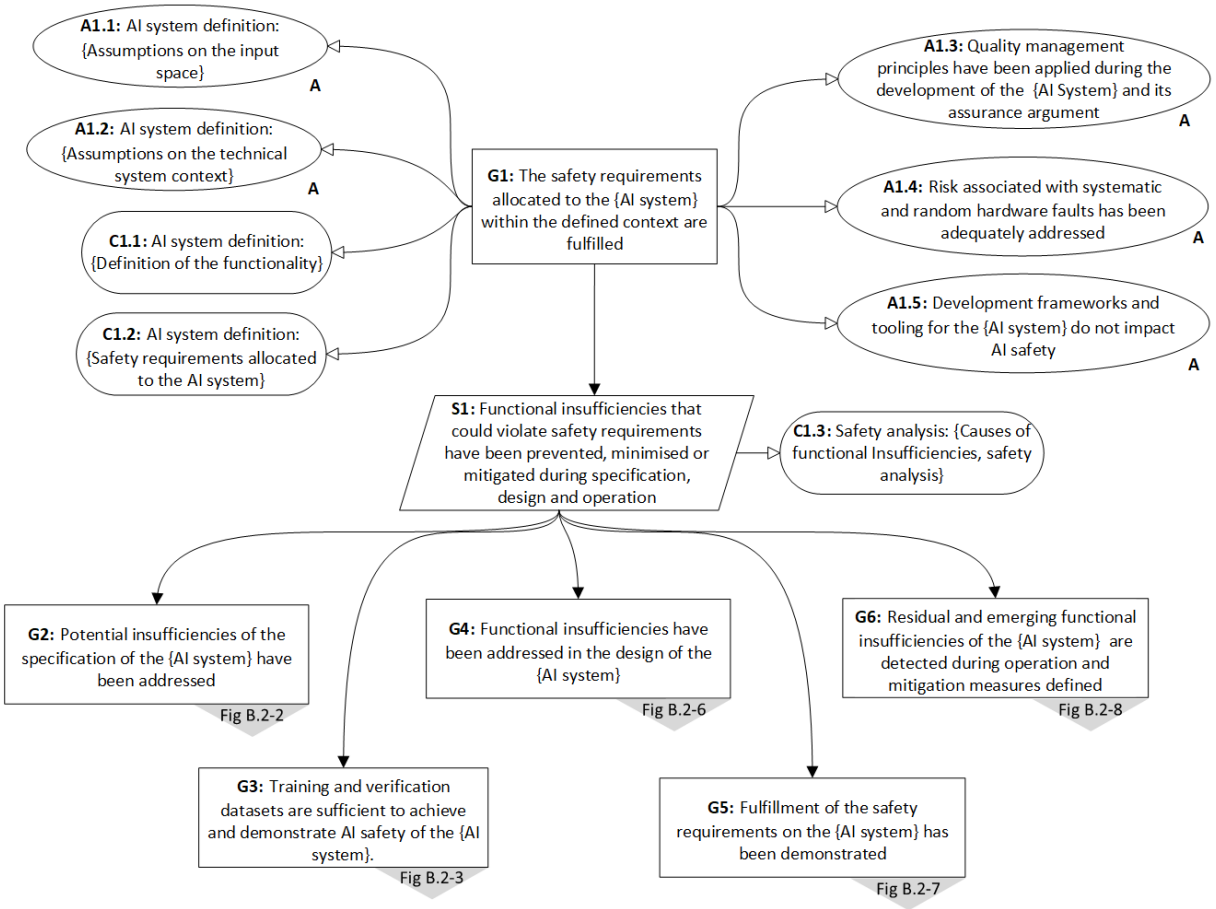


Figure B.2–1 — Assurance argument pattern for a supervised machine learning-based AI system

Figure B.2–2 elaborates the claim G2 of the argument pattern that demonstrates that potential insufficiencies of the specification have been addressed as described in Clause 9. This argument pattern consists of demonstrating:

— A sufficient understanding of the input space (G2.1);

— The derived AI safety requirements are complete and consistent with respect to the safety requirements allocated to the AI system. This includes demonstrating that each individual AI safety requirement is well defined on the basis of safety-related properties of ML models (S2.2) as well as that the combination of all derived AI safety requirements are sufficient to fulfil the safety requirements allocated to the AI system (G2.2.1);

— The performance limitations of the AI system are sufficiently well defined that a safe behaviour at the system level can be ensured (G2.3).

© ISO 2024 – All rights reserved

The derivation of the AI safety requirements as well as the definition of residual performance limitations are supported by the use of safety analyses (see Clause 13) that determine the potential for safety-related functional insufficiencies and potential causes in the AI system.

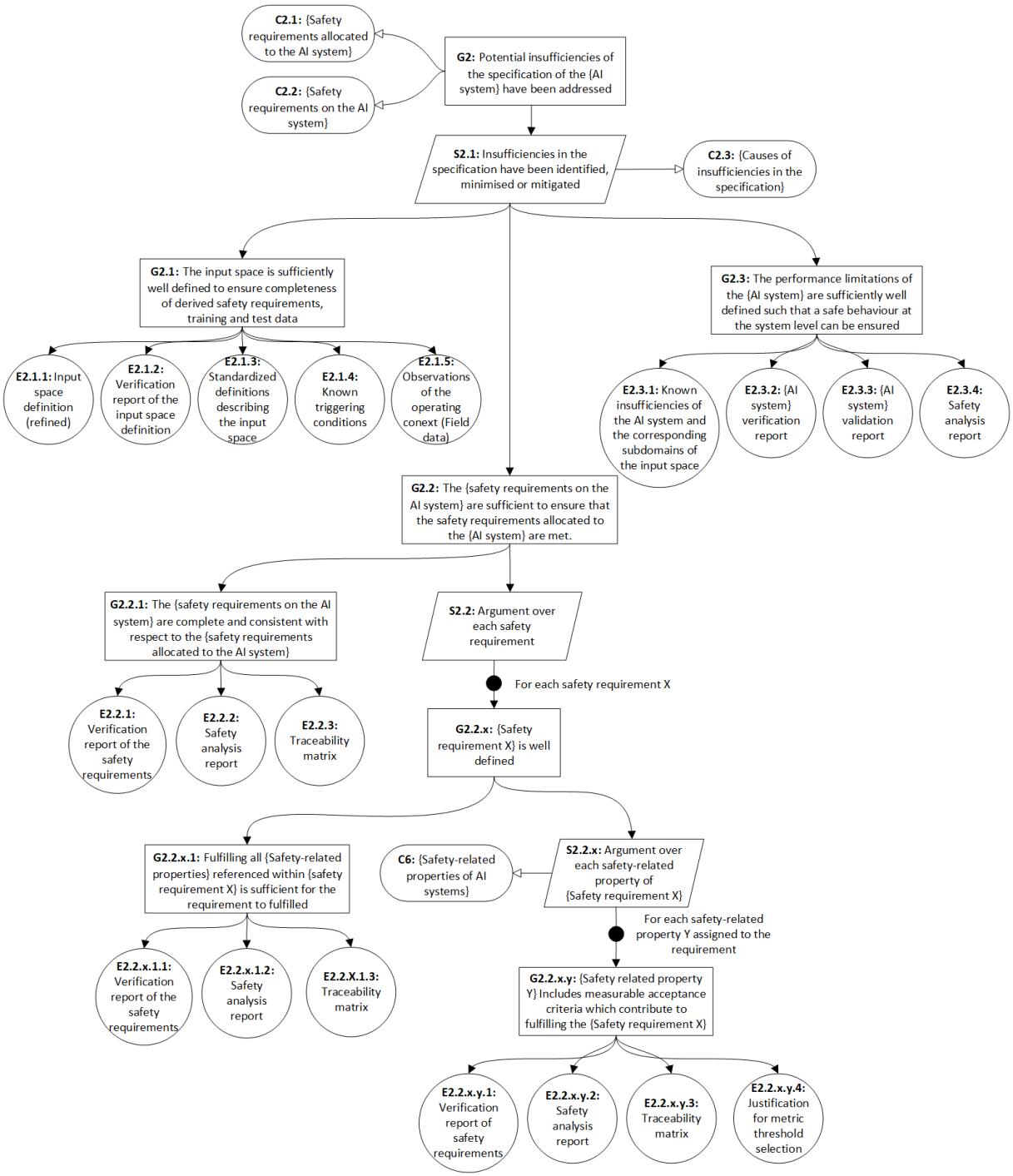


Figure B.2–2 — Assurance argument pattern for demonstrating potential insufficiencies of the specification of the AI system have been addressed

Figure B.2–3, Figure B.2–4 and Figure B.2–5elaborate the claim G3 of the argument pattern that demonstrates that the datasets used for training and verification of the AI system are sufficient to achieve and demonstrate AI Safety, as described in Clause Clause 11. This claim is further refined as follows:

— The datasets consist of suitable selections of observations from the overall input space (G3.1);

© ISO 2024 – All rights reserved

— The integrity of the datasets is maintained throughout the data lifecycle (G3.2).

The assurance argument is supported by a set of safety-related properties of the datasets, which can be specific to the application and applied AI technology (see Clause 11.4.3.2 for examples).

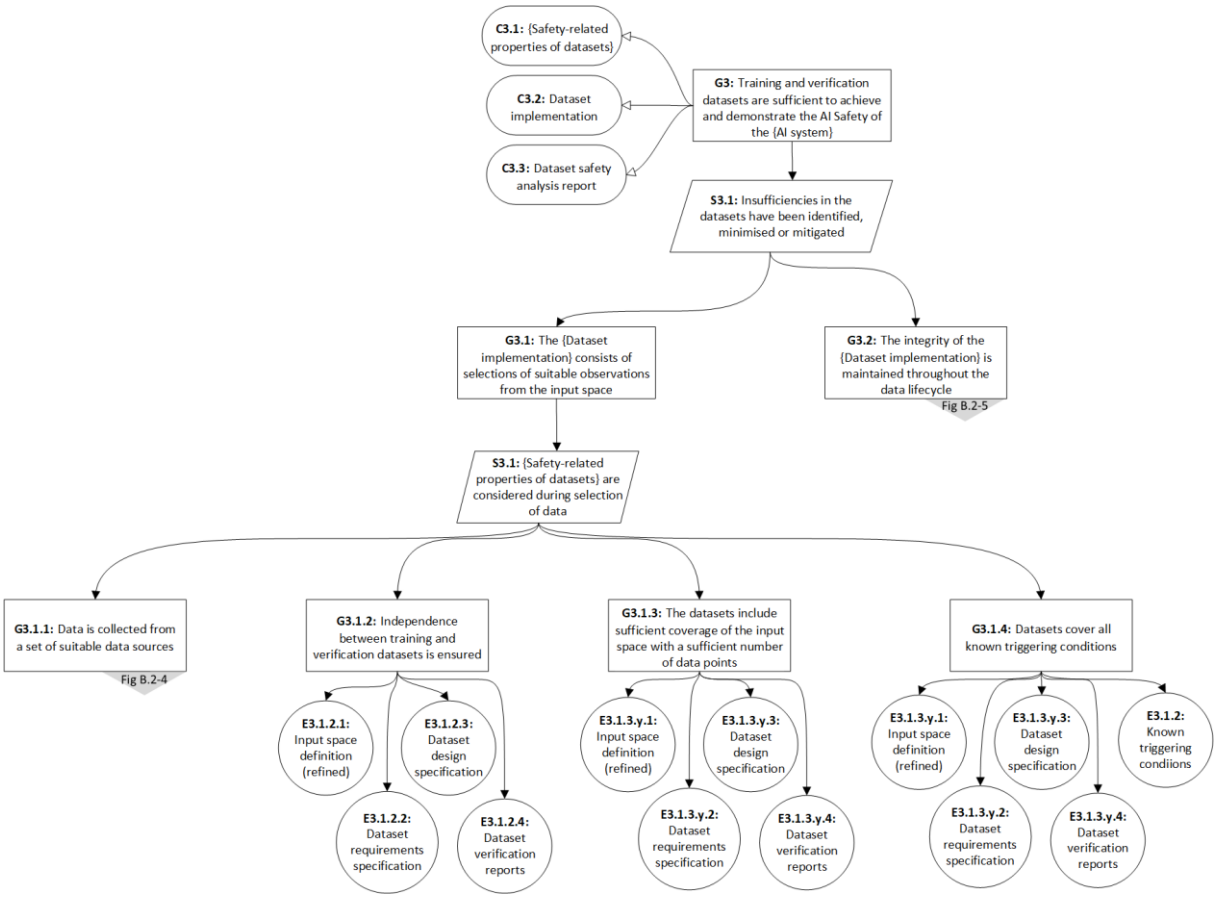


Figure B.2–3 — Assurance argument for the sufficiency of the datasets

© ISO 2024 – All rights reserved

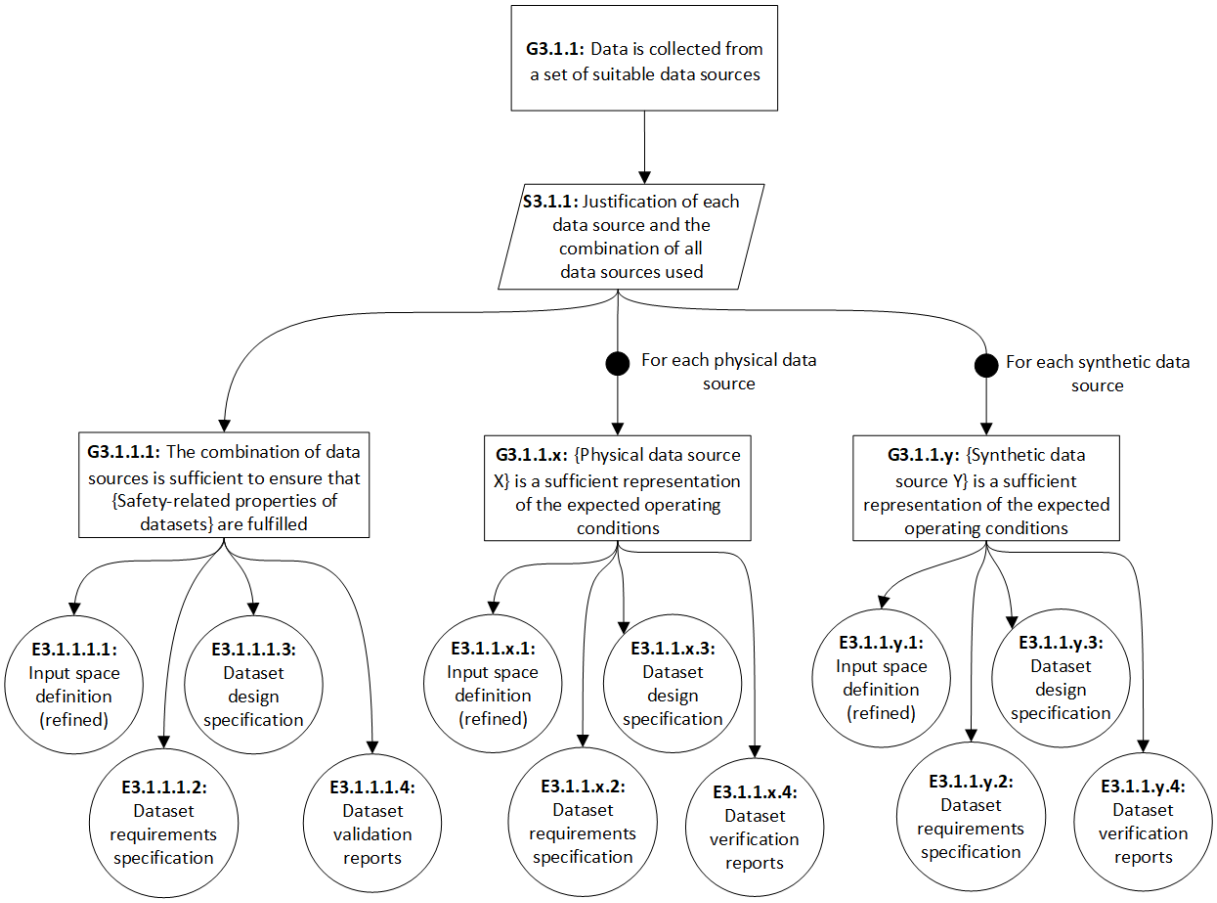


Figure B.2–4 — Assurance argument for claim G3.1.1

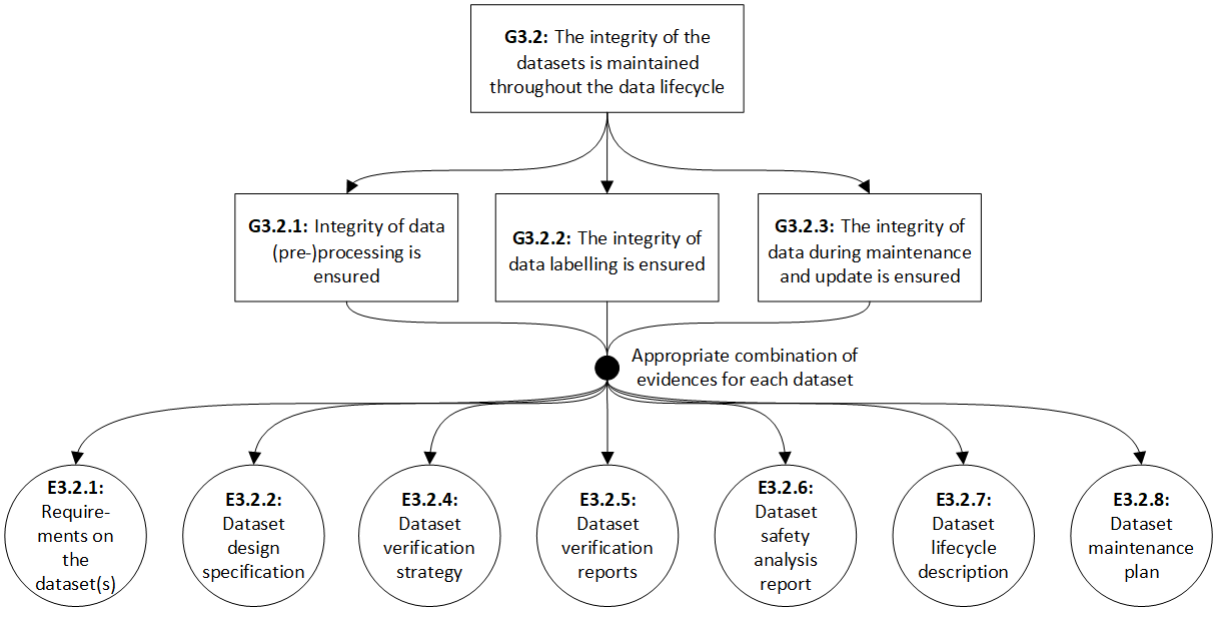


Figure B.2–5 — Assurance argument for claim G3.2

Figure B.2–6 elaborates the claim G4 that demonstrates that functional insufficiencies have been addressed in the design of the AI system, as described in Clause 10. This claim is made based on an understanding of the

© ISO 2024 – All rights reserved

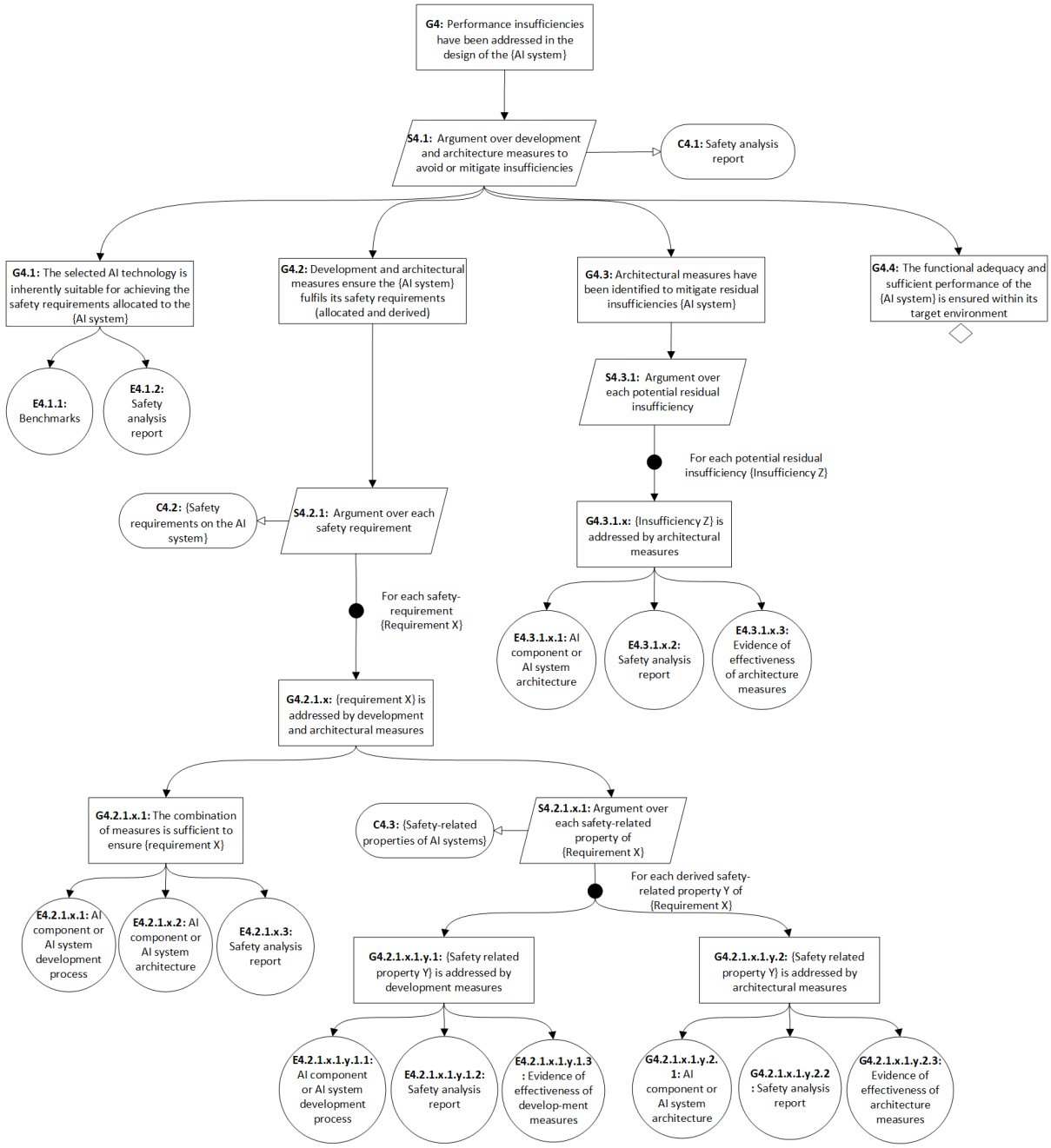
factors that influence the fulfilment of the AI safety requirements as well as the effectiveness of proposed development and architectural measures. The claim is further refined as follows:

— The chosen AI technology is inherently suitable for achieving the safety requirements allocated to the AI system (G4.1),

— Development and architectural measures have been chosen that ensure that the AI system meets its AI safety requirements (G4.2),

— Architectural measures have been identified to mitigate residual insufficiencies in the AI model (G4.3), and

— The functional adequacy and sufficient performance is also ensured within its target environment (G4.4). Note: G4.4 is not elaborated further in this version of the document.



© ISO 2024 – All rights reserved

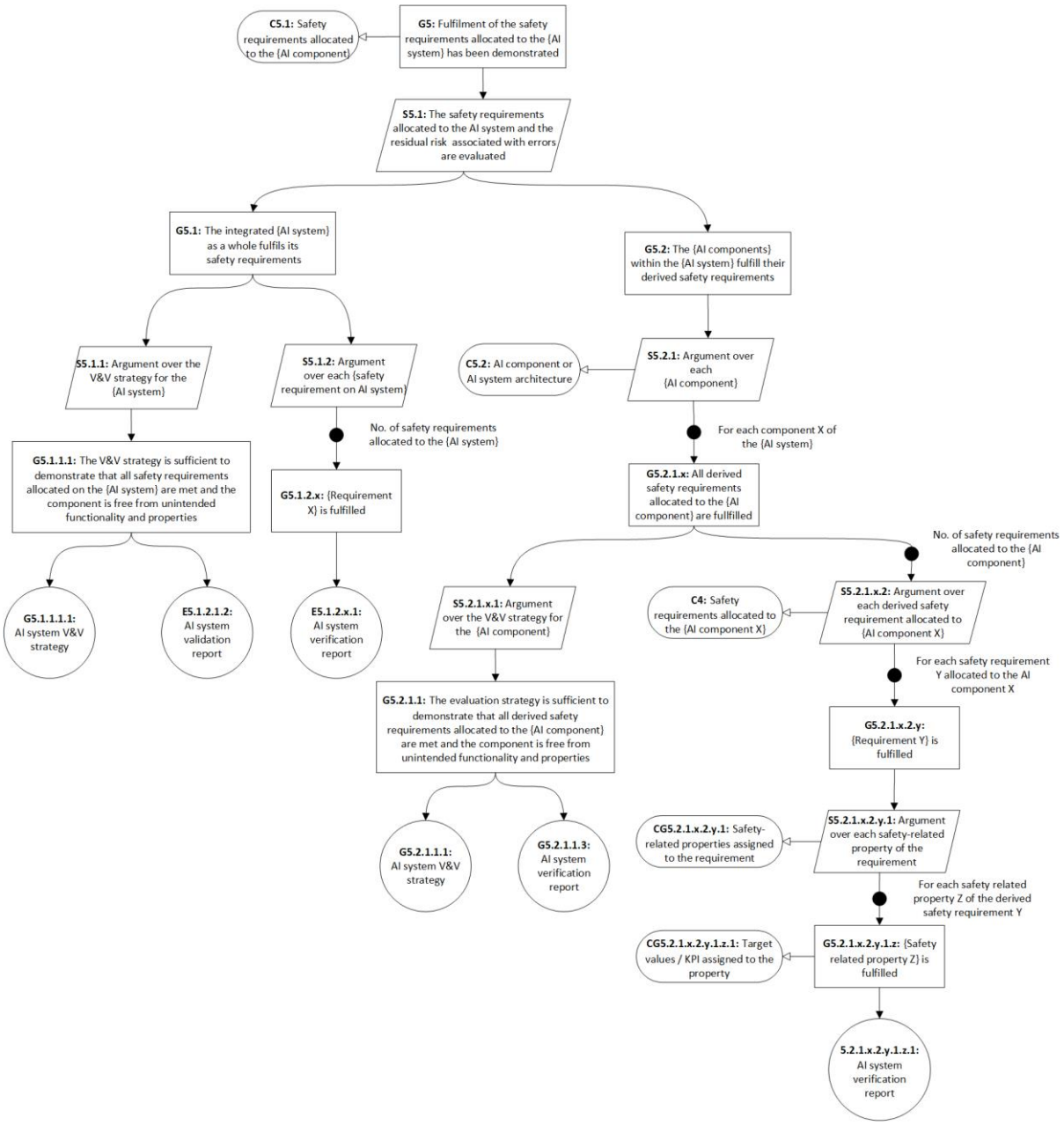
Figure B.2–6 — Assurance argument pattern that functional insufficiencies have been addressed during design

Figure B.2–7 elaborates the claim that sufficient evidence exits that the safety requirements allocated to the AI system have been fulfilled as demonstrated through verification and validation as described in Clause 12. This argument considers:

— The fulfilment of the safety requirements allocated to the AI system in itsentirety (G5.1), and

— The fulfilment of the derived AI safety requirements allocated to the individual components of the AI system (G5.2).

In each case, an argument is made over the appropriateness of the verification and validation strategy as well as the evidence used to evaluate each individual requirement.



© ISO 2024 – All rights reserved

Figure B.2–7 — Assurance argument pattern that the fulfilment of the safety requirements has been adequately demonstrated

Figure B.2–8 elaborates the claim that residual and emerging insufficiencies are identified during operation and mitigation measures are defined, as described in Clause 14. This argument considers:

— The definition of effective operating procedures for the safe operation of the encompassing system based on known insufficiencies of the AI system (G6.1),

— The use of effective processes for continuous re-evaluation of residual risk (G6.2), and

— Effective countermeasures are taken to address emerging insufficiencies (G6.3). This claim in the assurance argument can only be made after initial release of the AI system during re-evaluation of the overall safety assurance argument before deployment of changes.

In each case an argument is made over the effectiveness of the measures to control risk during operation.

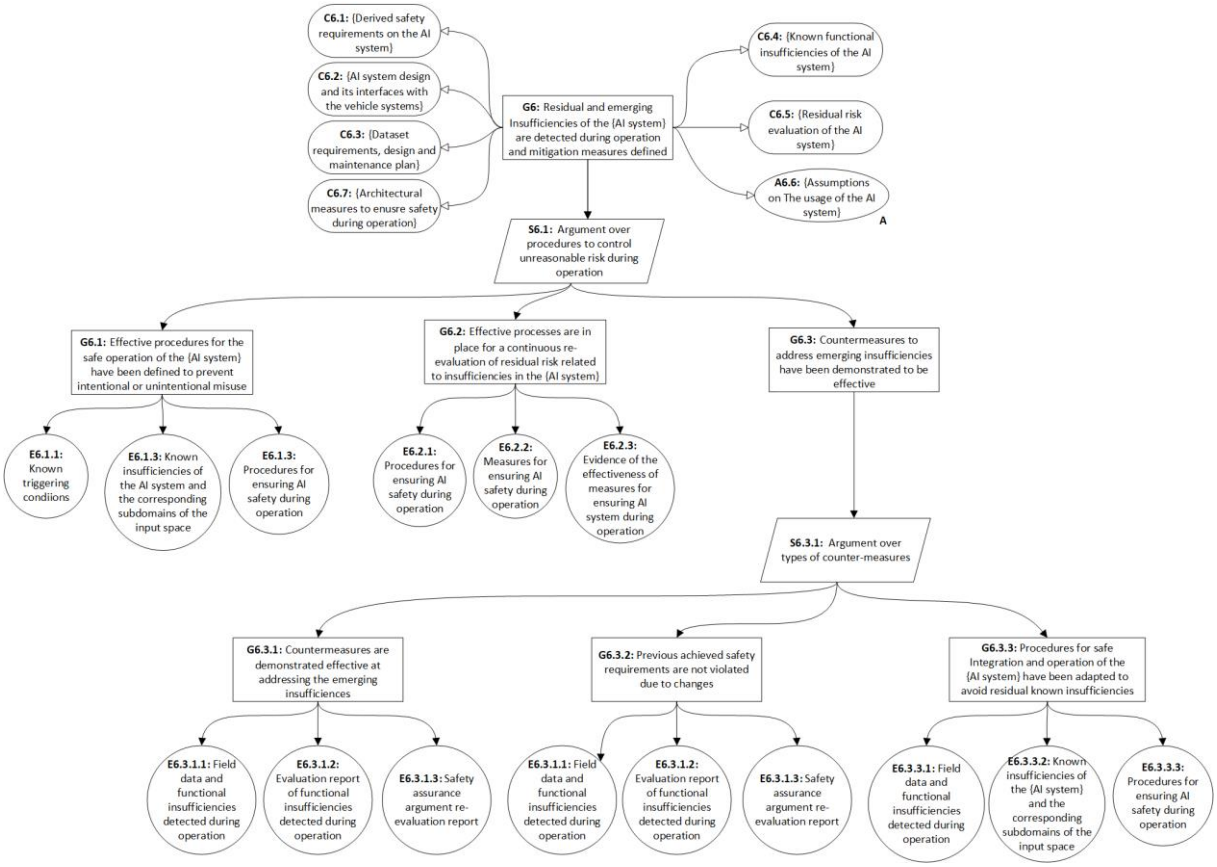


Figure B.2–8 — Assurance argument pattern that emerging and residual insufficiencies are identified and mitigated during operation

B.3 Use of assurance claim points to increase confidence in the assurance argument

B.3.1 General remarks on the use of assurance claim points

The GSN pattern outlined in Clause B.2 reflects the basic structure of an assurance argument that safety requirements allocated to a supervised ML-based AI system are fulfilled. The GSN pattern reflects the objectives and requirements of this document.

© ISO 2024 – All rights reserved

For any given AI system, the strength of the assurance argument may depend on a number of factors related to the complexity of the task and its environment, the availability of sufficient training and test data and the types of AI-technique used. These factors can lead to uncertainty and therefore diminished confidence in the argument.

As outlined in clause 8.7, an evaluation of the argument can include identification of defeaters based on the following types of assertions within the assurance argument [11]:

— **Asserted context**: relationship between the claim, contextual information and assumptions;

— **Asserted evidence**: relationship between the claim and the evidence supporting that claim;

— **Asserted inference**: relationship between the claim and the strategies used to structure the sub-claims and evidence to support that claim.

Version 3.0 of the GSN standard [6] provides the mechanism of assurance claim points (ACPs) to add further, deeper reasoning for particular relationships that would otherwise potentially undermine the confidence in the argument.

The reasoning linked to a particular ACP can be supplied in various forms. A separate GSN model for each ACP is one option, evaluation reports with links to further supporting evidence is another.

The following subclauses provide examples for ACPs to support each of the above types of assertion.

B.3.2 Example assurance claim points to support assumptions or context: ACP-A2 for assumption A2

Referring to Figure B.2–1, this subclauses addresses ACP-A2 related to the asserted context associated with ACP-A2 as illustrated in Figure B.3–1.

The assumption A1.2 reads “{Assumptions on the technical system context}”. This refers to the technical integration of the AI-system into the encompassing system, i. e. this assumption refers to the interfaces to the other systems and sub-systems as part of the vehicle.

EXAMPLE 1 ISO/IEC/IEEE 15289:2019 clause 10.28 mentions some of the properties which are subject to proper interface definitions: “systems or configuration items performing the interface (including human-system and human-human interfaces), standards and protocols, responsible parties, information or data records transmitted by the interface, interface operational schedule, and error handling”.

The information linked to ACP-A2 would demonstrate why the documented interface properties are considered complete in the sense that no safety-relevant interface property remains unspecified. I. e. none of the unspecified interface propertiesare able to interfere with the achievement of the safety requirements allocated to the AI-system.

EXAMPLE 2 For a camera-based object detection and classification function implemented by an AI system, information regarding the resolution, depth of focus, quality (e.g. sensor noise), etc., of the camera providing the raw images is documented and analysed to ensure that it meets the assumptions made during the development and test of the AI system.

© ISO 2024 – All rights reserved

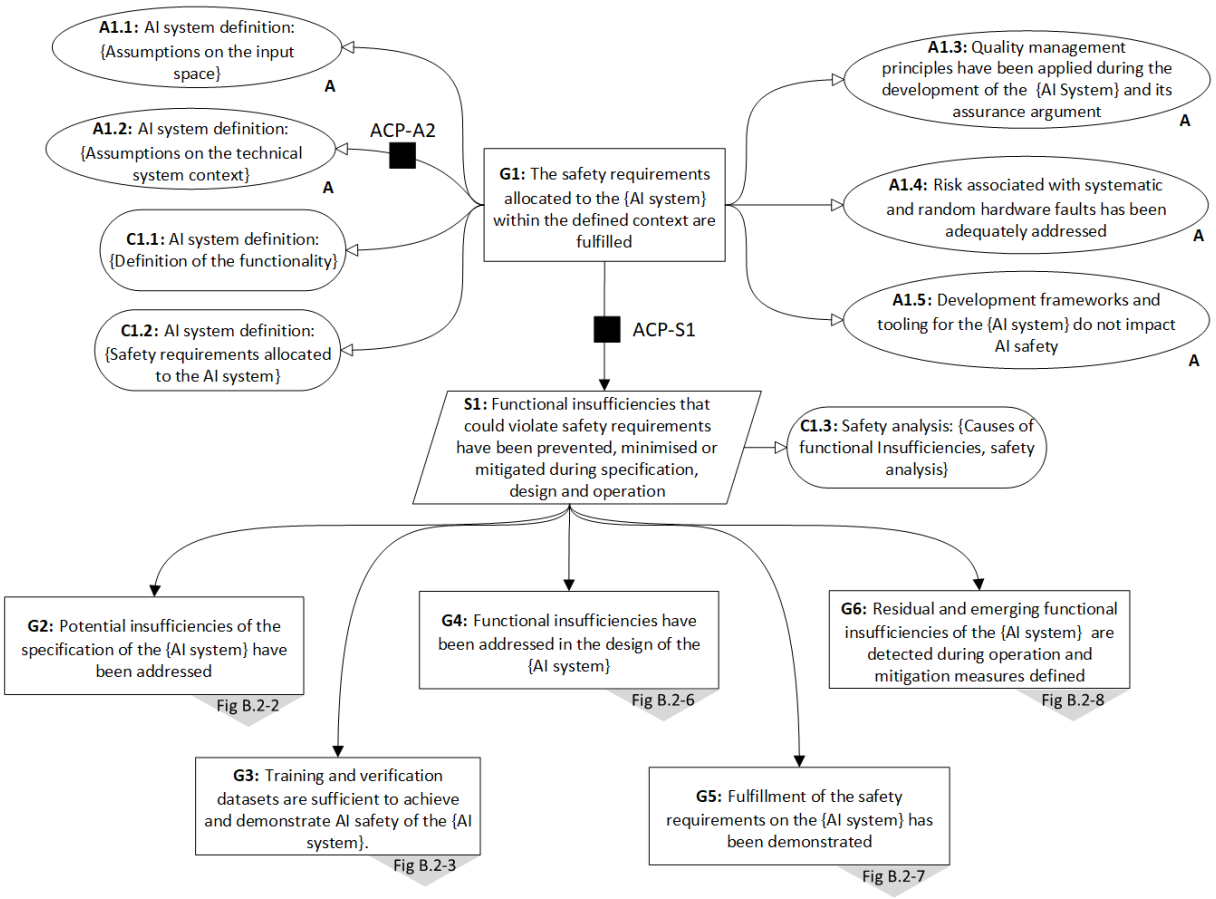


Figure B.3–1 — Example use of ACPs within the GSN assurance argument pattern

B.3.3 Example assurance claim point to support inference: ACP-S1 for strategy S1

The strategy S1 reads “Functional insufficiencies that could violate safety requirements have been prevented, minimized or mitigated during specification, design and operation”. S1 is supported by sub-goals which reflect the various clauses of this document. The ACP-S1 in Figure B.3–1 is inserted in order to strengthen the assertion that the argumentation strategy based on the set of hypothesized causes of insufficiencies derived from safety analysis is complete and sufficient to demonstrate that the safety requirements allocated to the AI system have been met.

This could be achieved by providing further information on previously (successful) applications of the strategy. Alternatively, this could be achieved by provisioning an evaluation procedure that supervises the strategy and would flag slipped, untreated functional insufficiencies. Additional forms of reasoning could include reference to effectiveness of the safety analysis approach (seeClause 13) to identify potential insufficiencies and their causes that may otherwise not have come to light during the development and test of the AI system.

B.3.4 Example assurance claim point to support evidence: ACP-E5

This subclause provides an example of an assurance claim point to support the assertion directly related to evidence. Figure B.3–2 illustrates the ACP-E5 in the context of the GSN provided previously in Figure B.2–7. This portion of the argument pattern relates to how a combination of evidence demonstrates that individual safety requirements are met.

Evidenceis asserted to show the achievement of eachAI safety requirement. Such evidence might be a collection of test results documented in one or more test reports. In the case that the safety requirement, test results and test reports are closely aligned with each other, no additional argumentation may be required.

© ISO 2024 – All rights reserved

However, in some cases the alignment between test cases and requirement might need additional justification. In other cases, the challenges might be associated with the testability of the requirement itself. Additional reasoning, such as traceability from requirements to test cases, or a description of the approach to indirectly verify a requirement, might be added to the argument using an assurance claim point such as ACP-E5.

This additional reasoning may address both the integrity of the evidence (e.g. have the results of the tests been collected and analysed without loss of critical information) as well as its validity (e.g. have sufficient tests been performed to ensure a high level of statistical confidence).

As in the previous subclauses, the assurance claim point can be linked to yet another separate GSN or be backed by some argument in natural language or other supportinganalyses.

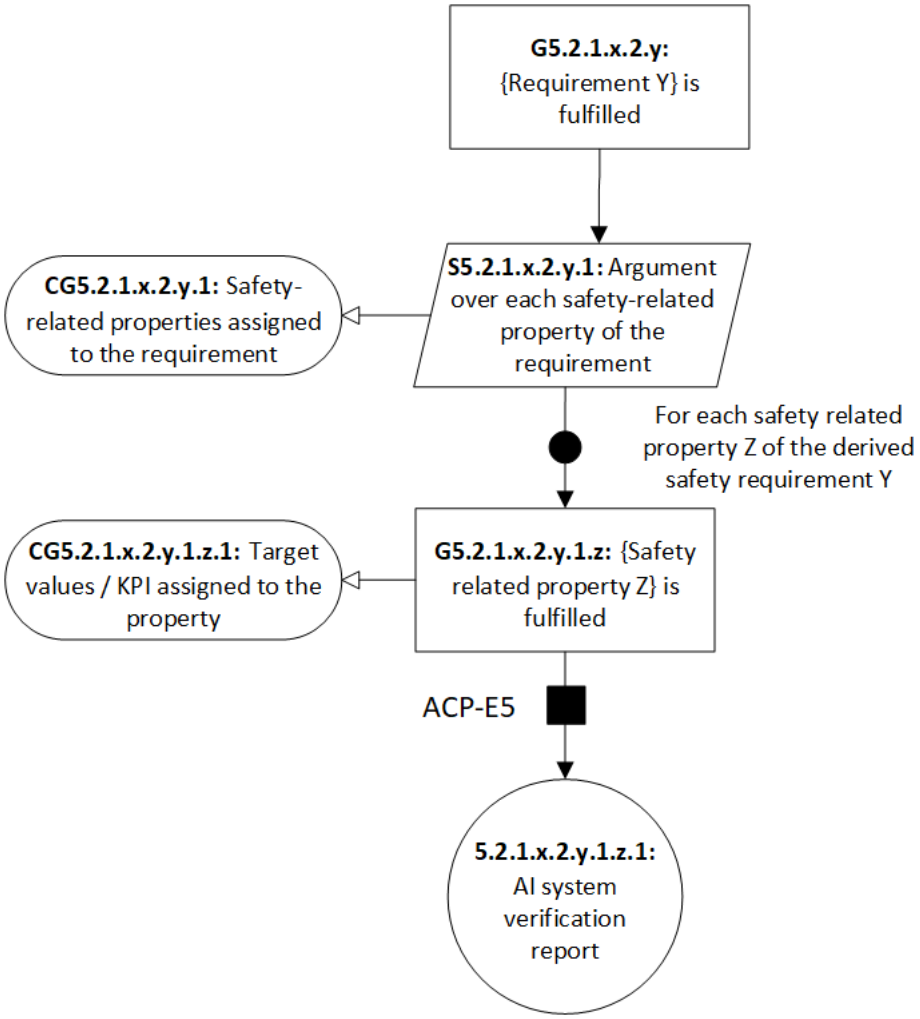


Figure B.3–2 — Example use of ACP to reason about the asserted evidence

© ISO 2024 – All rights reserved

Annex C

ISO 26262:2018 Gap Analysis for ML

This annex presents the results of a gap analysis of ISO 26262:2018 related to ML. The gap analysis is in the form of an example tailoring and guidance for ISO 26262:2018 Parts 4 and 6. The analysis did not find significant gaps related to ISO 26262:2018 Parts 1,2,3,5,7,8,9.

C.1 ISO 26262-4:2018 Tailoring and Guidance for ML

Table C.1–1 presents the example tailoring and guidance for the requirements of ISO 26262-4:2018, Clause 8 "Safety Validation" related to ML. The requirements of ISO 26262-4:2018 and requirements from Clause 8 that are not listed in the table are considered to not need any additional tailoring or guidance for ML. Different tailoring can be applied to different AI technologies.

Table C.1–1 — ISO 26262-4:2018, Clause 8 example tailoring/guidance for ML

Clause	Requirement	**Proposed Tailoring/Guidance for ML**
8.4.3.3	The safety validation at the vehicle level, based on the safety goals, the functional safety requirements and the intended use, shall be executed as planned using: a) the safety validation procedures and test cases for each safety goal including detailed pass/fail criteria; and b) the scope of application. This may include issues such as configuration, environmental conditions, driving situations, operational use cases, etc.	Additional guidance: intended use includes representative inputspace definition (e.g. Operating environment, input domain, conditions of use) a) explicitly includes safety-related KPIs
8.4.3.4	An appropriate set of the following methods shall be applied: a) repeatable tests with specified test procedures, test cases, and pass/fail criteria; b) analyses; c) long-term tests, such as vehicle driving schedules and captured test fleets; d) operational use cases under real-life conditions, panel or blind tests, or expert panels; and e) reviews.	Guidance b) may be limited (e.g. simulation only) e) is typically not applicable for ML validation

C.2 ISO 26262-6:2018 Tailoring for ML

© ISO 2024 – All rights reserved

Table C.2–1 presents the example tailoring and guidance for the requirements of ISO 26262-6:2018 related to ML. Where noted, the tailoring/guidance is related to NN models only. The requirements of ISO 26262-6:2018 that are not listed in the table are considered not to need any additional tailoring or guidance for ML.

Table C.2–1 — ISO 26262-6:2018 example tailoring/guidance for ML

Clause	Requirement/Method	**Proposed Tailoring/Guidance for ML**
5.4.3 Table 1	1a Enforcement of low complexity 1b Use of language subsets 1c Enforcement of strong typing 1d Use of defensive implementation techniques 1e Use of well-trusted design principles 1f Use of unambiguous graphical representation 1g Use of style guides 1h Use of naming conventions 1i Concurrency aspects	Tailoring For ML applications, Table 1 applies unchanged for use case independent elements (e.g. CUDA C++ libraries). For the use case dependent elements (i.e. the models), 1c), 1d), 1g), 1i) with “o” for all ASILs (justification see ISO/IEC TR 5469 Tables A.3 and A.4) NOTE Use case independent elements refer to elements that behave the same independent of the use case (i.e., CUDA libaries fulfill the same purpose independently if the trained models use case is in autonomous driving or predictive maintanance). In contrast, use case dependent elements like neural network models are dependent on the specific use case, which changes their properties.
6.4.1	The software safety requirements shall be derived considering the required safety-related functionalities and properties of the software, whose failures could lead to the violation of a technical safety requirement allocated to software	Guidance in the form of additional considerations Requirements in the form of: a) KPIs; b) data attributes; and c) dataset requirements (e.g. inputspace definition) specification for the training/validation/testing data set The ML implementation not meeting its KPIs is an ISO 26262 issue. Incorrect or insufficient KPIs are a SOTIF concern.
6.4.4	The hardware-software interface specification initiated in ISO 26262-4:2018, Clause 6, shall be refined sufficiently to allow for the correct control and usage of the hardware by the software, and shall describe each safety related dependency between hardware and software	Guidance in the form of examples EXAMPLE 1 Software is specified to run on one CPU on a multi-CPU system. EXAMPLE 2 NN specified to run on a GPU
6.4.7	The software safety requirements and the refined requirements of the hardware-software interface specification shall be verified in accordance with ISO 26262-8:2018,	Guidance in the form of additional considerations e) adequate coverage of the input space of the software.

© ISO 2024 – All rights reserved

Clause	Requirement/Method	**Proposed Tailoring/Guidance for ML**
	Clauses 6 and 9, to provide evidence for their: a) suitability for software development; b) compliance and consistency with the technical safety requirements; c) compliance with the system design; and d) consistency with the hardware-software interface.	Adequate coverage of the input space typically involves i) sufficient labels that comprehend the entire labelling space (e.g. labels for emergency vehicles), ii) data with multiple views (e.g. multiple examples of emergency vehicles) f) address the handling of out-of-distribution inputs
7.4.3	In order to avoid systematic faults, the software architectural design shall exhibit the following characteristics by use of the principles listed in Table 3: a) comprehensibility; b) consistency; c) simplicity; d) verifiability; e) modularity; f) encapsulation; and g) maintainability.	Guidance 1) Software components implemented using machine learning are considered to be difficult to verify. A heuristic component is preferred over a machine learning component assuming the function can acceptably be implemented using a heuristic component. 2) NN models are considered as individual units. The principles of Table 3 typically cannot be met for NN applications. Usually, they are generated from higher level languages using tools. The design principles therefore are applied to the code that generates the NN model and tool qualification are applied to the generator. This is similar to the usage of code generation in normal SW development.
7.4.4	The software architectural design shall be developed down to the level where the software units are identified.	Guidance 1) An individual NN may consist of many nodes and layers but is typically considered as one unit. It may not be possible to express an NN software design at any level lower than the individual NN level. 2) A SW unit can be an NN so long as suitable interfaces can be defined and requirements allocated to those units. 3) An architecture description for an NN model, e.g. in ONNX, can be created. Nevertheless, explainability based on the architecture may be low however a justification for the choice of the structure can be provided, e.g. motivated by ablation studies.

© ISO 2024 – All rights reserved

Clause	Requirement/Method	**Proposed Tailoring/Guidance for ML**
7.4.7	If a pre-existing software architectural element is used without modifications in order to meet the assigned safety requirements without being developed according to the ISO 26262 series of standards, then it shall be qualified in accordance with ISO 26262-8:2018, Clause 12.	Guidance For pre-existing ML based software when the specification characteristics such as dataset attributes, KPIs, inputspace definition and output metrics are articulated, the verification should ensure that the specification characteristics are sufficiently met.
7.4.13	An upper estimation of required resources for the embedded software shall be made, including: a) the execution time; b) the storage space; and c) the communication resources.	Guidance in the form of additional considerations d) parallel computation resources
8.4.3	To avoid systematic faults and to ensure that the software unit design achieves the following properties, the software unit design shall be described using the notations listed in Table 5. a) consistency; b) comprehensibility; c) maintainability; and d) verifiability.	Guidance in the form of additional considerations Additionally, use the derived AI safety-related properties for the given AI system as appropriate, for example: e) Interpretability (ISO/IEC 5469 for definition); f) Explainability (see Annex D and ISO/IEC 5469 for definition); g) Predictability (see Annex Dfor definition); h) Specificability (see ISO/IEC 5469 for definition); i) Generalisation (see Annex D and ISO/IEC 5469 for definition); j) Domain shift (see ISO/IEC 5469 for definition); k) Robustness-Safeness (see ISO/IEC 5469 for definition); m) Diversity (see ISO/IEC 5469 for definition); and n) Confidence (see ISO/IEC 5469 for definition).
8.4.4	The specification of the software units shall describe the functional behaviour and the internal design to the level of detail necessary for their implementation.	Guidance For the case of a unit containing an NN, the structure of the NN (e.g. number of nodes, layout, interconnects and activation function) and hyperparameters and training methods of NN (e.g learning rate) are part of the specification of the software unit.
8.4.5	Design principles for software unit design and implementation at the	Guidance

© ISO 2024 – All rights reserved

Clause	Requirement/Method	**Proposed Tailoring/Guidance for ML**
	source code level as listed in Table 6 shall be applied to achieve the following properties: a) correct order of execution of subprograms and functions within the software units, based on the software architectural design; b) consistency of the interfaces between the software units; c) correctness of data flow and control flow between and within the software units; d) simplicity; e) readability and comprehensibility; f) robustness; g) suitability for software modification; and h) verifiability.	1) For NN units, a) and c) may not apply since the NN is considered as one function and the order of execution of individual nodes is not guaranteed. 2) For h), the structure of the network can be verified, for example, by inspection that the correct structure is implemented. 3) For AI models trained using data, the influencing factors of Table 9–1 may be considered as additional design principles: observation certainty, label certainty, model certainty, and operation certainty
9.4.2	The software unit design and the implemented software unit shall be verified in accordance with ISO 26262-8:2018, Clause 9 by applying an appropriate combination of methods according to Table 7 to provide evidence for: a) compliance with the requirements regarding the unit design and implementation in accordance with Clause 8; b) the compliance of the source code with its design specification; c) compliance with the specification of the hardware-software interface (in accordance with 6.4.4), if applicable; d) confidence in the absence of unintended functionality and properties; e) sufficient resources to support their functionality and properties; and f) implementation of the safety measures resulting from the safety-oriented analyses in accordance with 7.4.10 and 7.4.11.	Guidance a) The NN model software verification report documents the test result KPI and the dataset used for the testing. Tailor d) to d) confidence in the absence of unintended functionality and properties (Unintended functionality is primarily a SOTIF concern for systems modelled using ML).
9.4.2 Table 7	1a Walkthrough 1b Pair-programming 1c Inspection 1d Semi-formal verification 1e Formal verification	Tailoring 1a) through 1i) For ML "o" since often infeasible to do effectively Guidance

© ISO 2024 – All rights reserved

Clause	Requirement/Method	**Proposed Tailoring/Guidance for ML**
	1f Control flow analysis 1g Data flow analysis 1h Static code analysis 1i Static analyses based on abstract interpretation 1j Requirement-based test 1k Interface test 1l Fault injection test 1m Resource usage evaluation 1n Back-to-back test between model and code, if applicable	1) For ML, 1j feasible for only some properties such as invariants and equivariants 2) For ML, 1l Fault injection has limited applicability 3) For ML, 1n Applicable when comparing off-line versus optimized code versions
9.4.3	To enable the specification of appropriate test cases for the software unit testing in accordance with 9.4.2, test cases shall be derived using the methods as listed in Table 8.	Guidance For ML, unit test cases can be selected from test dataset
9.4.4	To evaluate the completeness of verification and to provide evidence that the objectives for unit testing are adequately achieved, the coverage of requirements at the software unit level shall be determined and the structural coverage shall be measured in accordance with the metrics as listed in Table 9. If the achieved structural coverage is considered insufficient, either additional test cases shall be specified or a rationale based on other methods shall be provided.	Tailoring Requirement NA for NNs, includes NA for Table 9 For NNs without separate program statements, branches or decision logic this requirement does not apply . An example of where this requirement is still applicable is conditional computation in neural networks, which is sometimes implemented to reduce latency and save energy (i.e., only part of a net is activated). It is possible to select inputs as unit tests to cover all branches.
9.4.5	The test environment for software unit testing shall be suitable for achieving the objectives of the unit testing considering the target environment. If the software unit testing is not carried out in the target environment, the differences in the source and object code, as well as the differences between the test environment and the target environment, shall be analysed in order to specify additional tests in the target environment during the subsequent test phases.	Guidance in the form of an example EXAMPLE A NN model is trained using fp32 math, but the on-line inferencing uses int8 for throughput and bandwidth savings. Off-line unit testing of the model uses int8 to match the target environment.
10.4.2 Table 10	1a Requirements-based test 1b Interface test 1c Fault injection test	Tailoring 1c) For NNs, targeted SW fault injection might only be appropriate at certain interfaces. HW fault

© ISO 2024 – All rights reserved

Clause	Requirement/Method	**Proposed Tailoring/Guidance for ML**
	1d Resource usage evaluation 1e Back-to-back test between model and code, if applicable 1f Verification of the control and data flow 1g Static code analysis 1h Static analyses based on abstract interpretation	injection on the target environment can test the response to permanent and transient faults. 1g) For NNs "o" 1h) For NNs "o" Justification g) and h) Static code analysis aiming to verify functionality of NN does not scale beyond small networks

3769

© ISO 2024 – All rights reserved

Annex D

Detailed considerations on safety-related properties of AI systems

This Annex provides a list of properties of AI systems that are considered desirable/necessary from a safety perspective. These properties are conceptual, and the list is based on past AI development experience and is not exhaustive.

Safety-related properties can be quantitative in nature as well as qualitative. As a result, they are not always completely achievable. For example, while the robustness property indicates that a model is either robust or not, a DNN model for classifying objects in an open world is never 100% robust against all possible insignificant input changes. The choice of safety-related properties relevant to the AI system should be validated through safety analysis to ensure their contribution to the system's safety, and target thresholds should be provided with justification.

A safety-related property may or may not apply depending on use cases, systems, AI models, etc. For example, while a self-driving vehicle's actions, such as acceleration and steering, could be controllable, the outputs of a DNN model for object detection in the perception pipeline of the vehicle are not.

The scope of a safety-related property of AI systems refers to the entity to which the property is attributed. For example, the organization can effectively and efficiently update an AI model whenever necessary. In this context, the overall system is considered the whole product, e.g., the vehicle.

Table D–1 — Safety-related properties of AI systems

NOTE 1 One or more KPIs are typically defined to characterize each safety-related property. The safety requirement specifies the acceptable threshold value for these KPIs.

Property	**Description**	Scope
AI robustness	Ability to maintain an acceptable level of performance under the presence of semantically insignificant, but reasonably expected changes to the input (see definition *AI reliability* (3.1.14)) NOTE 2 AI robustness focuses on foreseeable/relevant perturbations (type of perturbation as well as amplitude of perturbation) which can occur in the real world, to avoid defining unnecessary safety requirements.	Model, system
AI generalization capability	Ability of a model to adapt and perform well on the previously unseen data during inference	Model, system
AI reliability	Ability to maintain all functionalities for a specified period (see definition *AI reliability* (3.1.12))	Model, system
AI resilience	Ability to quickly recover from an incident (see definition *AI resilience* (3.1.13))	(Overall) system, organization

© ISO 2024 – All rights reserved

Property	**Description**	Scope
AI controllability	Ability of an external agent to overwrite the behaviour or output of an AI system	(Overall) system, organization
AI explainability	Ability to explain in natural language which factors influence the AI element decision and how (see definition *AI explainability* (3.1.4))	Process
AI predictability	Reliable confidence information for AI refers to the ability of an AI model to reliably indicate if its prediction can be trusted or not. This is not always true for all kinds of models. For example, the output of a softmax function is frequently misinterpreted as some kind of posterior distribution which indicates confidence.	Model, system
AI alignment	AI alignment is ensuring that the AI system behaviour is aligned with human values and with the human expected intent of the system	Process
Justified design decisions	Bad design decisions may have detrimental effects on the behavior of the AI model/system. Therefore, design decisions need to be justified and their negative effects need to be analyzed. This is also valid for training process and data selection decisions where applicable.	Process
Maintainability	Ability to effectively and efficiently identify operational insufficiencies and countermeasures, change the encompassing system design and the AI system design, collect relevant data, label them, train the AI model, and update the AI system and other systems in a timely manner.	Organization, process
AI bias and fairness	AI bias refers to the notion that an AI model or dataset maybe systematically prejudiced towards some kind of (potentially erroneous) assumption. This assumption stems from the inherent statistical distributions (e.g., over classes) in a dataset that can be learned by a model. If the model bias is linked to a difference in treatment of certain subgroups of humans (e.g., ethnic minorities, age or sex) this model is considered unfair. AI fairness is the reasonable absence of unfairness.	Model, (overall) system
Distributional shift over time	Distributional shift over time refers to the potential distributional change	Overall system

© ISO 2024 – All rights reserved

Property	Description	Scope
	in any input data stream due to natural changes (e.g., sensor aging, new object classes on streets, …). This distributional shift can cause a performance decrease of the AI model in the field since this model has been developed with a different data distribution.	

© ISO 2024 – All rights reserved

Annex E

STAMP/STPA example

E.1 Overview

Clause E.2 describes a vulnerable road user (VRU) recognition and braking system as an informative example. An example of a Vulnerable Road User (VRU) is a pedestrian whose trajectory intersects with, and moves toward, the path of the ego vehicle during nighttime. The system identifies VRUs and, depending on the situations, sends a brake command to either reduce speed to a certain extent or to stop completely. E.2.4 demonstrates how to identify causes of safety-related errors of the recognize of pedestrians. E.2.5 focuses on deriving safety measures to mitigate the safety risks due to the safety-related AI errors. These mitigation measures for the AI system can be categorized into those applied during design-time and those applied during operation-time [51].

NOTE The term "ego vehicle" in this example is used for the vehicle fitted with functionality that is being analysed for the SOTIF [Source : ISO 21448 3.6 ego vehicle].

E.2 STPA Example

E.2.1 STPA Step 1: Defining the purpose and scope of the analysis

The first step of STPA identifies the stake holders' losses to be prevented. Table E.2–1 provides an example of STPA losses and hazards.

Table E.2–1 — Example of loss and hazard identification

Loss	Hazard
[L1] Loss of life or human harm (severe or fatal injuries)	[H1] Vehicle violates minimum distance threshold/ requirement from/with vulnerable road users.

E.2.2 STPA step 2: Modelling of the control structure

The development and operation of AI systems involve various complexities and uncertainties in the assurance, such as data quality, training, complexity of AI models, etc., which potentially correlate to or influence safety-related AI errors.

NOTE Control structures are defined as hierarchical structures where each level imposes constraints on the activities of the level beneath them and accidents are viewed as the consequence of inadequate control of safety constraints [52].

Figure E.2–1 illustrates the control structure of the VRU recognition and braking system utilizing bounding boxes or sematic segmentations. The AI system receives the camera or the LiDAR data from the sensors to recognises the VRU.

© ISO 2024 – All rights reserved

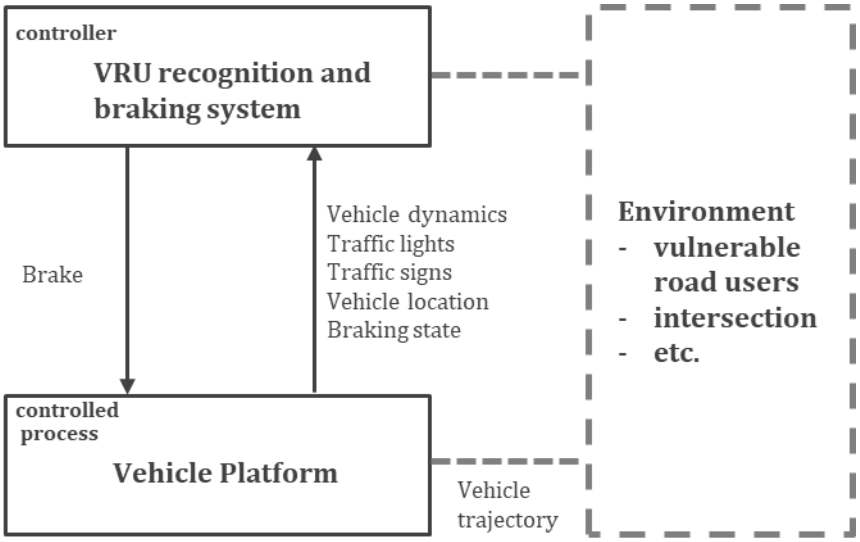


Figure E.2–1 — Control structure of the VRU recognition and braking system

E.2.3 STPA step 3: Identification of unsafe control actions

Table E.2–2 shows a few examples of the unsafe control actions (UCAs) of the AI system when sending the brake command to actuators. These UCAs are control actions which in a particular context and operational situation can lead to a hazard.

NOTE A control action is defined as a command or feedback item created by, or used by, a system element to perform its function(s) [53].

Table E.2–2 — Example of unsafe control actions for the control action "AI system output state"

Control action	Not providing	Providing	Providing too early, too late, or in the wrong order	Providing for too long or stopping too soon
CA1: AI system sends the brake command	UCA1: AI system does not send the brake command when the VRU is approaching to the trajectory of the ego vehicle. [H1] (False negative)	UCA2: AI system send the brake command when there are no VRU in the trajectory of the ego vehicle. [xx] (False positive)	UCA3: AI system sends the brake command too late when the VRU is moving in the trajectory of the ego vehicle. [H1]	

E.2.4 STPA step 4: Identification of causal scenarios

In STPA Step 4, the focus is to identify the causal scenarios that could lead to the occurrence of each UCA and the failure of control actions to be executed or executed correctly. Figure E.2–2 shows detailed control structure focusing on control loops and interactions.

© ISO 2024 – All rights reserved

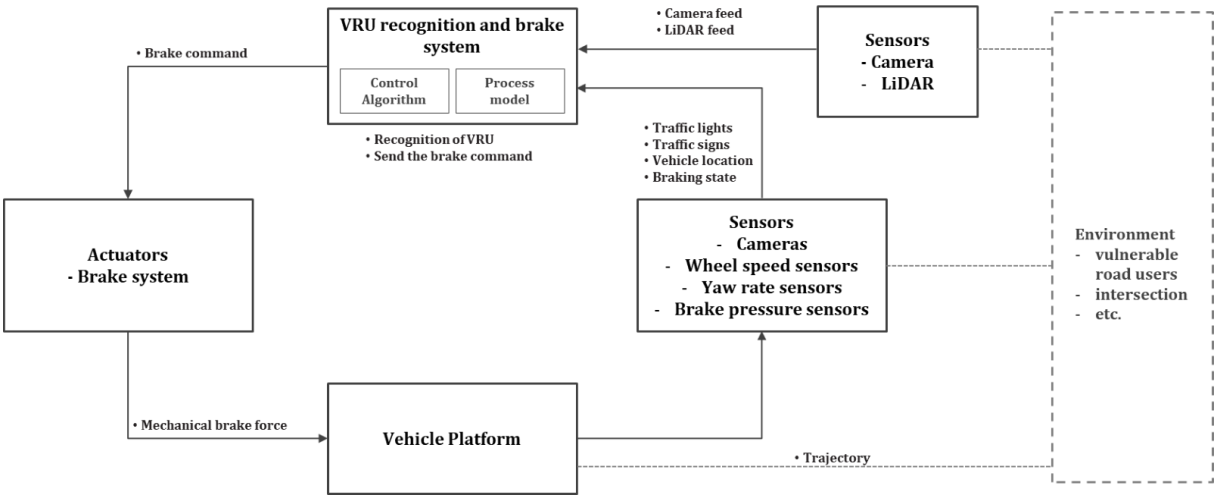


Figure E.2–2 — Detailed control structure of the VRU recognition and braking system

Table E.2–3 shows some examples of causal scenarios which are correlated or influenced by using influencing factors defined in Clause 6 (Table E.2–4).

NOTE The example presented here addresses only a few causal scenarios that lead to UCA1. However, in an STPA, all UCAs need to be analysed completely.

UCA1: AI system does not send the brake command when the VRU is approaching to the trajectory of ego vehicle. [H1] (False negative)

Table E.2–3 — Example of causal scenarios of safety-related AI errors

Process model or control algorithm	Category of causal scenarios	Causal scenarios	Influencing factors (Link to countermeasure)
Process model	Ontological uncertainty of recognizing VRU	A vulnerable road user, dressed in unusual clothing and walking in a peculiar posture at the intersection, was a scenario that had not been previously trained for and evaluated during the development, resulting in a high rate of false negatives.	Observation
Process model	Epistemic uncertainty of recognizing VRU	Steam rising from the surface obscures vulnerable road users (VRUs) and distorts the images captured, leading to AI errors such as misplacement or incorrect sizing of bounding boxes, resulting in some objects being missed or mislabeled. (Bounding box use case)	Operation, Label
Process model	Aleatoric uncertainty of recognizing VRU in specific	A pedestrian unexpectedly reverses direction while still looking forward,	Model

© ISO 2024 – All rights reserved

	behaviors and appearances	resulting in an increased rate of AI errors in predicting the pedestrian's trajectory.	
Process model and Control algorithm	Aleatoric uncertainty of recognizing VRU in environmental noise and performance of resources	Dust and moisture, which can accumulate on the surface of the sensors, are captured as homogeneous noise in the images. This leads to a higher increase in computational demand than originally designed for, along with lowered prediction scores, ultimately resulting in incorrect semantic segmentation beyond the required time value. (semantic segmentation use case)	Operation, Model

By illustrating these correlations or influencing factors, it is possible to develop effective safety measures such as design-time measures and/or operation-time measures [51] that can prevent or mitigate the occurrence of UCA.

E.2.5 Identifying safety measures to mitigate the safety-related issues

After identifying safety-related issues in an AI system, the next step is to incorporate safety measures into the design, dataset generation and implementation of the AI system according to Clause 9, Clause 10 and Clause 11 of this document.

Table E.2–4 shows examples of chain of the safety related issues of AI system and safety measures.

Table E.2–4 — Examples of the chain of the safety related issues of the AI system and safety measures [51], [54].

Chain of the safety related issues			Safety measure			
AI Error class	**Insufficiency**	**Cause**	**Design-time measure**	**Metric**	**Operation-time measure**	**Metric**
Observation (Incorrect classification)	Lack of generalization	Under specification, scalable over-sight	Balanced training set	Coverage of the ODD model	N/A	N/A
Observation (Incorrect classification)	Unreliable confidence values	Overconfidence due to uncalibrated soft-max values	Temperature scaling	Remaining AI error rate, remaining accuracy rate	N/A	N/A
...	...	...	...	...	...	...
Operation (False negatives)	Lack of robustness	Instability of DNNs for minor changes to the inputs	Adversarial training	Adversarial and perturbation robustness	Robustness certificates	Certifiable perturbation strength

© ISO 2024 – All rights reserved

Operation (Sequence of false negatives)	Lack of generalization	Under specification, scalable oversight	Balanced training set	Coverage of the ODD model	Comparison with other sensor data	Diagnostic coverage
...	...	...	...	...	...	...
Observation (False positives)	Clever Hans effect	Spurious correlations in the training data	Diversified training set	Conceptual disentanglement	Plausibility checks	Diagnostic coverage
Operation (False positives)	Lack of generalization	Distributional shift	N/A	N/A	Out of distribution detection	Diagnostic coverage
Label						
...	...	...	...	...	...	...
Model						
...	...	...	...	...	...	...

© ISO 2024 – All rights reserved

Annex F

Identification of software units within NN-based systems

ISO 26262 defines the concept of a SW unit as comprising of its interface, and SW architecture designs. This approach is central to developing a modular process for software development and verification and validation (V&V). Some benefits of a modular software unit are:

— With clearly identified software units and their interfaces, detecting the negative impact of one software unit modification on the other software units can be anticipated and mitigated early on.

— Modularity improves V&V and testing, allowing an exponential reduction in the complexity of the number and size of test inputs to achieve comparable coverage [64] and also improved fault localization.

— Modularity also allows for incremental V&V and testing of individual software and hardware elements without waiting for complete system integration.

AI models are increasingly used to implement complex functionality, although the notion of software units for some AI methods (e.g. CNNs) has not been clearly defined. Applying the concept of software units to AI models therefore presents certain challenges:

— While the design of AI models is routinely specified using neural network architectures understood as computational graphs and NN layers, individual neurons and layers are not separately testable units.

— The size and complexity of groups of neurons and layers alone are not valid criteria to distinguish them as modular units.

NOTE 1 For conventional algorithms, software units are typically identified based on the size, complexity, and relatedness of implementation artefacts; each software unit's implementation is kept to a reasonable size and complexity, and preferably represent only one function's implementation.

— In contrast to conventional software, any function implemented by a NN, especially a monolithic one, might not be clearly mappable to a subset of the network with a clear boundary, and the allocation to individual neurons and layers evolves as the NN is re-trained.

This annex describes a method to identify SW units within a given NN-based system, and their organization into a SW architecture. While monolithic NN designs are possible, the intent of the method is to encourage dividing a NN-based system performing complex functions into separate elements that can be tested or analysed more easily.

The key aspects to defining a SW unit are:

— A clearly defined interface;

— A clearly defined function;

— The ability to perform V&V activities at the unit interface level, such as unit testing and inspections.

In contrast to conventional SW units, different parts of an NN, such as NN layers, often exchange representations that are latent and also evolve as the NN is re-trained. A latent representation is the task-specific information extracted from the input and mapped into a latent vector or tensor space that aids performing the task.

© ISO 2024 – All rights reserved

EXAMPLE 1 In image classification, features extracted from images that belong to the same class are located in the latent space close to each other to aid their subsequent classification.

The mapping into the latent representations and the representations themselves normally evolve as the NN is re-trained. Further, latent representations might not be interpretable by humans, but it might be possible to map them into interpretable representations using suitable decoders.

NOTE 2 A latent feature is a property of a model that captures underlying characteristics or patterns in the data. Latent features are inferred during a model's learning process, and contribute to a models ability to represent complex relationships within the data. Latent features can also be used to demonstrate compliance with AI safety requirements, please refer to G.4.8

As opposed to conventional SW architectures where the information passed through the interface is defined and fixed, in a NN architecture, the information available at the interface might change, for example the number of object classes, but the intent of the interface can remain unchanged, e.g. visualizing a classification.

Locations in an NN where latent representations are exchanged and can also be used for V&V activities, become candidates for SW unit interfaces within the NN-based system. These locations might require some form of decoding of the latent representation into one amenable to these V&V activities. Examples of such V&V activities at these points include:

— Assessing the performance of an NN-based element; for example, by comparing the NN-based element output with ground truth data.

— Assessing the level of uncertainty within the NN-based element; for example, an epistemic uncertainty could be assessed looking at the output distribution of an NN ensemble.

— Assessing the plausibility of the representations; for example, physical objects are expected to respect spaciotemporal consistency.

— Providing visualization for human inspections; for example, semantic segmentation, instance segmentation, panoptic segmentation;

— Obtaining some insight on the interpretability of an NN-based element; for example visualizing the attention of an NN-based element within its input to produce its output.

NOTE 3 Semantic Segmentation is the process of assigning a label to every pixel in the image. This is in stark contrast to classification. Instance segmentation is the task of detecting and delineating each distinct object of interest appearing in an image. Panoptic segmentation unifies two distinct concepts used to segment images namely, semantic segmentation and instance segmentation.

Once candidate interfaces are determined, potential SW subsystems and SW units within the NN-based system can be identified. These SW subsystems and SW units can be organized hierarchically into a SW architecture. A key guiding principle is to perform this decomposition with respect to system functions, where the atomic elements are SW units with still clearly assigned functional responsibilities. In contrast to SW units in conventional software, NN-based SW units can be trained end-to-end, which can evolve the latent representations passed at the interfaces.

— The developer can check whether or not all features are correctly identified therefore using the NN output as a visualization point; this defines an interface which intent is to visualize detected features.

— Initially the NN can be trained to detect n features. At a later stage the NN can be re-trained to detect n+1 features. The information passed from this NN to the fusion NN has changed, from n to n+1 features, but the intent, the visualization of the features, has not changed.

In summary, the concept of an ML element, which is either a NN-based SW unit or subsystem has these properties:

© ISO 2024 – All rights reserved

a) the function represented by the NN-based element can be swapped or composed with other functions to implement a higher-level function.

b) As per ISO 26262 definition of “SW unit”, the NN-based element can be subjected to unit-level V&V analyses, such as testing and inspection, separately.

EXAMPLE 2 As part of a sensor fusion architecture, combining LiDAR and map-based input signals, the NN-based element used to create a latent representation of LiDAR features can be defined as a SW unit. The LiDAR features output of this element could be compared, via a suitable decoder like an object detector head, with ground truth data. The NN-based element performing the conversion between the Voxelized LiDAR data and the LiDAR features data can therefore be tested in isolation from the other architecture’s elements.

— It is worth noting that with NN-based architecture, not all SW units are testable in isolation. Instead, some SW units will be tested separately, while others will be tested after integration with other units. For example, when the input of a SW unit is a complex representation, it might not be efficient to emulate that representation for the sake of unit testing as opposed to testing that unit when integrated with the unit(s) providing its input representation.

© ISO 2024 – All rights reserved

Annex G

Architectural and Development Measures for AI Systems

G.1 Examples of architectural and development measures for AI systems

A variety of AI system development and architectural measures exist, are being improved, and new ones are emerging in this rapidly evolving field. The decision about what measures can be used to develop an AI system and their impacts on safety cannot be generally assessed in such a diverse context.

Evidence to justify the selection of development and architectural measures can include, amongst others, benchmarks, qualitative and quantitative safety analyses, ablation studies or analytical arguments related to intrinsic properties of the AI models (e.g., capability for generalization, explainability, robustness, transparency, maintainability etc.). Architectural decisions can be documented by means of strategy used, solution and evidence to support the decision.

The following sub-clauses provide details on the measures captured in Table 7–1.

G.1.1 Measures for Architectural Redundancy

ISO/TR 5469, Figure 5, describes some architectural redundancy patterns for systems using AI technology components. Figure G.1–1 translates them in the context of the reference architecture for an AI system as represented in Figure 6–7.

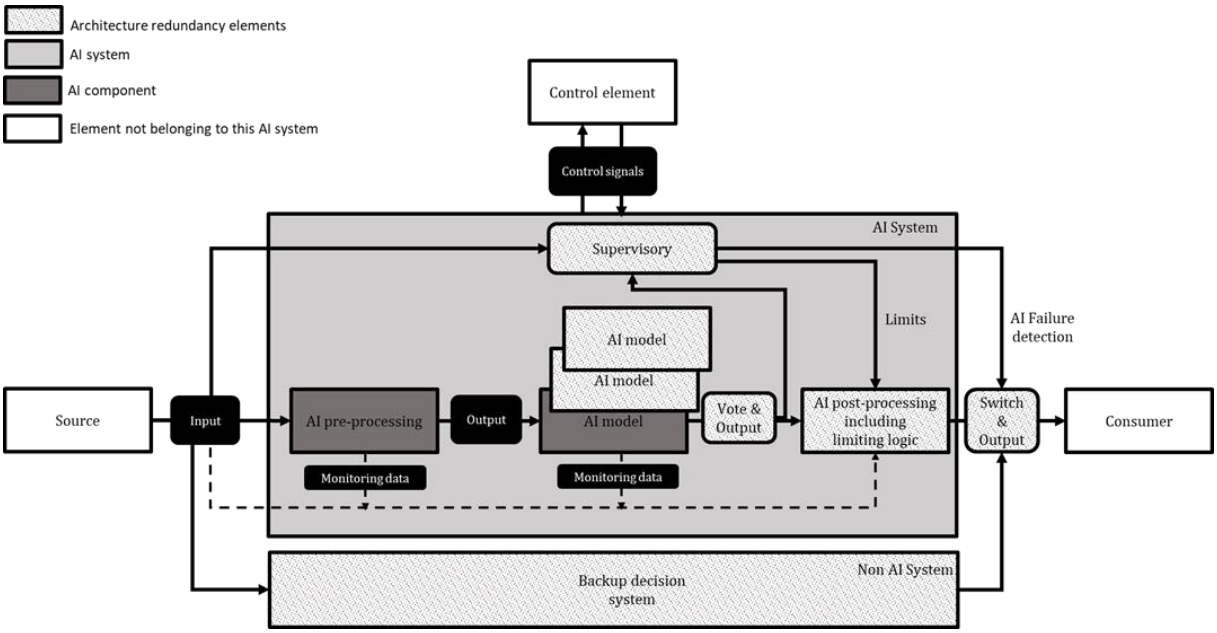


Figure G.1–1 — Architectural redundancy patterns for AI system

Some of the possible architectural redundancy patterns are:

— Use of redundant and diverse AI models, including ensembles

— Use of N-version programming of AI models

— Use of supervisory and limiting logic

© ISO 2024 – All rights reserved

Usage of non-AI models as backup decision system.

NOTE 1 Depending on the use case, one or more architectural redundancy patterns are used

NOTE 2 DNN models could include redundancy and early fusion within themselves such that it is not necessary to have late fusion on complete models. In some cases, this could lead to better nominal performance.

The architectural redundancy is expected to either enhance the AI properties such as robustness, resilience, etc, or to add different features or functionalities or to ensure that the failure of the AI model is detected and mitigated. If for the detection of the failure of an AI model another AI model is used then, in order to achieve the required independence, diversity can be necessary. The following subclauses provides more details on each of the listed architectural redundancy patterns.

G.1.1.1 Diverse redundant models

Redundancy can be achieved by voting of diverse models. It involves combining multiple AI technologies fulfilling the same functionality, but implemented starting from different problem formulations, using different training data or different models.

EXAMPLE There can be a set of AI models that are used for dynamic and static object detection and classification, lane detection, and path/trajectory planning and another end-to-end AI models) that can be based on behavioural cloning [55]. In this case the diversity is qualitatively easier to argue as the stated AI models are based of different dataset, different input and outputs and different network architecture. The AI models on one side are learned to identify specific obstacles and then do the specific path/trajectory planning to avoid them whereas the end-to-end AI model(s) on the other side are learned to do the end-to-end task by identifying the drivable area without necessarily identifying the obstacles or lanes. It is likely that there is no full equivalence/overlap of ability/performance; in such a case additional AI or non-AI models can be added to overcome the deficiency.

G.1.1.2 Model ensembles

Ensemble methods combine the predictions obtained by multiple models to create a consolidated output [56]. The multiple models can be of different architecture and have different hyperparameters or of similar architecture but trained on different data sets. These methods have been successfully employed for improving accuracy in object detection tasks. Various forms of ensemble methods exist. A single model, multiple data ensemble method [57] proposes to use data augmentation for creating multiple inputs, and uses a fuzzy integral method to combine the output across these inputs. The same model is used across different inputs. Ensembles can be achieved by using various methods including but not limited to: bagging, boosting, random Forest, gradient boosting, and stacking.

G.1.1.3 N-version diverse programming

In this method, multiple independent versions of an AI model that are built to predict the same output when the same input is provided [58]. The independence and diversity are targeted to be attained via using different training data, different AI model architectures or different training process. Some averaging or majority voting is then performed to select a more robust prediction. The objective is to achieve fewer common errors across the multiple versions of the models. The independence and diversity can also be targeted by using different model input (e.g. two cameras with different angle) [59]. N-version can also be achieved by using the same base model and using different dropouts creating different models. N-version programming can improve the fault tolerance and reliability of AI models.

G.1.1.4 Supervisory, limiting logic and non-AI backup system

It is possible that an AI system can be constrained to work within a predefined safe envelope. Safe limits require that a subset of the action space (safe envelope) can be determined and are minimally restrictive on safe AI components' behaviour. Simple limits on an AI model's output(s)can result in the AI model to mimic the limiter itself therefore negating the benefit. This subsystem architecture is sometime referred to as a safety cage, which enforces behaviour onto the subsystem. Different types of online monitoring of AI System can be implemented in the supervisor and limiting logic, such as uncertainty modelling and out-of-distribution

© ISO 2024 – All rights reserved

detection [60]. Safety monitoring is also a well-known dependability technique. This approach is generally based on a system model or the environment and on properties they are designed to guarantee [61].

G.1.1.5 Selection techniques for architectural redundancy (voting and switching)

With architectural redundancy patterns, there is the need of a decision procedure to compute the final result based on the outputs of redundant models. A simple decision procedure could be voting-based. Taking AI-based classifiers as example, different voting schemes are common:In hard voting (also known as majority voting), every individual classifier votes for a class, and the majority wins. In statistical terms, the predicted target label of the ensemble is the mode of the distribution of individually predicted labels. In soft voting, every individual classifier provides a probability value that a specific data point belongs to a particular target class. The predictions are weighted by the classifier's importance and summed up. Then the target label with the greatest sum of weighted probabilities wins the vote. Switching is another approach where the selection of the base predictor model is made based on predefined rules. Two types of switching strategies are common:Threshold switching: model whose predictions satisfy certain conditions or a set of performance thresholds. Typically for this case the various models have different strengths and can therefore satisfy selection criteria differently depending on the input space. Performance switching: best performing model is selected.

G.1.1.6 Usage of “AI model” and “conventional software”

Components using conventional software can be used to perform the plausibility checks or verification of the output generated by the AI components. They can be also used if they provide redundant functionality (i.e., same objectives are also met, e.g., classical computer vision that doesn’t involve AI). In both the cases the intent is to detect the errors of the AI components. Switching to a system based on conventional software is a fallback possibility in case of error detection related to AI model.

G.2 Qualitative and quantitative analysis of AI architectures

This clause describes aspects related to a systematic development process of the AI component. A rigorous and systematic development process provides evidence that activities during development lead to safe outcomes and that AI work products have been analysed for systematic faults. For example:

— hyperparameters can be checked by peer reviews.

— it can be checked whether the AI architecture meets the AI safety requirements, or whether individual elements violate AI safety requirements allocated to them.

— an analysis of the AI architecture can be used to identify failures in the system at interpretable interfaces, and incorporate appropriate mechanisms for monitoring.

To support those activities, it is necessary to give preference to AI architectures with transparent interfaces between their AI components and interpretable representations so that errors become measurable.

EXAMPLE 1 For DNNs, the computational graph provides a reasonable opportunity for analysis, because the framework generates it automatically and it can be considered as an application graph. Object detection models, including many Faster R-CNN based ones, include operations on lists of objects in the application graph. Their graph contains interpretable information about sets of object bounding boxes, which are refined in several steps. Potential objects can be discarded by an operation that limits the number of elements in the list. In the worst case, e.g., this results in a pedestrian not being detected due to this operation. The impact of the limitation operation on safety requirements is considered because it is a potentially systematic fault in the design.

Architecture analyses aiming to discover the potential for contributing AI errors will investigate the tolerance, adaptability and information flow of the proposed architectural concept. Investigation into the tolerance of the architectural construct will provide arguments and evidence addressing at least the following:

— how failures of sub systems that provide inputs are tolerated,

© ISO 2024 – All rights reserved

— how the concept processes inputs inconsistent with existing training, test and validation data,

— how faults and failures internal to the computation are tolerated,

— how adversarial attempts to disrupt the computation are tolerated and

— how incorrect computation outputs are tolerated.

Investigation into the adaptation of the architectural construct will provide arguments and evidences addressing at least the following:

— how the architecture concept prevents or mitigates unauthorised adaptations,

— computational behaviour is assured before, during and after an adaptation and

— how new requirements are fulfilled.

Investigation into the information flow of the architectural construct will provide arguments and evidences addressing at least the following:

— information between sub-systems and interacting systems,

— how information is available to support maintenance and future development and

— how information is structured and stored to support real-time and post-incident analysis.

This allows potential safety-related failures to be identified. Nevertheless, failures are tolerable within certain limits. However, the occurrence of the failures depends on the specific trained DNN components. The safety analyses are supported by further measurements of a trained DNN, which quantify the concrete occurrence of failures. Based on this measurement appropriate safety measures can be derived. Suitable safety analysis techniques, their assumption, advantages and limitations are discussed in the **Table 13–1**. For a sufficiently described AI component, the individual elements can be systematically analysed. In particular, the description can represent AI-specific aspects, such as data representations using tensors in the case of DNNs, or the different development and operation phases of the AI components such as training or inference.

EXAMPLE 2 The hierarchical operations of the application graph can be analysed using the HAZOP method to a reasonable level in the hierarchy. For this purpose, the HAZOP guide words are interpreted accordingly, and the analysis examines if a deviation can lead to the violation of safety requirements. For example, the guide word “too large” can be associated with an element in the architecture such as an object bounding box, and the consequences are assessed to identify whether a safety requirement is violated. In a following step where the network is now trained, it can be observed how often the violation occurs on real data. If it is too often, additional measures can be taken iteratively to reduce the violation.

G.2.1 Identifying software units within AI architectures

ISO 26262 defines the concept of a SW unit its interface, and the associated SW architecture as central to the modular development and verification and validation (V&V) of software.

Annex F presents a method to decompose an AI architecture into elements that can be treated as software units similarly to the concept of software units presented in ISO 26262:2018 - part 6. In doing so, most of the software development measures specified in ISO 26262:2018 can be applied, with some tailoring *(*see Annex C), to gain confidence in the correctness of each software unit.

Additionally, each element of the AI architecture can be probed during development and forensic analyses and provide useful information such as confidence levels or representations of the AI element’s output for human inspection, helping in the resolution of insufficiencies.

© ISO 2024 – All rights reserved

G.3 Data distributions and their impacts on AI models

AI systems are prone to poor generalization when the distributions of the development data set (training and validation) differ from those of the test data set and/or the deployment (real life) data sets. There are typically two types of distribution related problems associated with AI models: distributional shift and out of distribution (OOD) samples. Distributional shift refers to changes within the same underlying distribution, while out-of-distribution data refers to data that is significantly different from the data the model was trained on. In both cases a previously well-trained and validated model might suffer from reduced generalization performance.

G.3.1 Out of distribution data and its mitigation

OOD data might cause the model to make incorrect predictions or be overly confident in its predictions. Examples of OOD samples in autonomous vehicles can include unique, unusual or unknown road signs, road marks, rare objects or scenarios not seen in the training data set. Architectural and development measures can help detect OOD and make the encompassing system robust to OOD data.

G.3.1.1 Architectural measures to address OOD robustness

Architectural measures to enhance robustness to OOD data include but are not limited to:

Ensemble models: combine the predictions of multiple models to make a final prediction. This can help to reduce the impact of individual models that might be sensitive to OOD data. Please refer to G.1.1.2 for details

Probabilistic Models: use probabilistic methods (such as Bayesian NNets) to model the predictive uncertainty of a model. This can help the model to better handle OOD data by being more cautious about making confident predictions on data that is different from the training data.

Domain Adaptation: involves adjusting the model architecture as needed to better reflect new features in the new domain.

Open set recognition: is a technique to enable models to be robust to unknown classes. These include measures such as novelty detection, outlier detection and threshold-based classification.

OOD Error Detection using reject function: is a technique where both in distribution and out of distribution data is considered as an input during the modified training step [62]. Furthermore, additional nodes as reject functions are used in the output layers of the AI model, this represents multiple reject classes.

NOTE All the OOD detection measures need to be designed in such a way that the OOD inputs are identified and rejected while meeting the relevant timing related safety requirements.

G.3.1.2 Development measures to address OOD robustness

Domain Adaptation: involves adapting the model to the new domain by re-weighting the loss function to better reflect new features in the new domain.

Data pre-processing: Pre-processing of data before training can improve model robustness. Some methods include methods such as normalization, feature scaling, outlier removal and data augmentation (adds diversity to data set).

Training measures: methods such as calibrating the prediction certainty of a model (see G.4.4), ensuring that the validation/test data sets are significantly different from the training set.

Adversarial training: these methods involve using adversarial training techniques to train the model on examples that are specifically designed to fool it and can also be used to establish OOD robustness. Examples include stickers on road speed limit signs.

G.3.2 Distributional shift and its mitigation

© ISO 2024 – All rights reserved

Distributional shift refers to changes in the distribution of the data that occur within the same underlying distribution. For example, if a model is trained on images of cars taken primarily during the day and then tested on images of cars taken only during night, this would be a distributional shift because the data distribution has changed within the same underlying distribution of cars but impacted by lighting conditions. One relevant example is the sudden appearance of face masks during the COVID pandemic. Face recognition algorithms were impacted by a shift in the distribution of faces. Distributional shift can lead to a drop in model performance because the model has not seen enough examples from the new distribution during training and might not generalize to the new data. In some cases, the shift might also occur from concurrently occurring effects such as for example low light and rain even though the model has been trained on each effect individually.

G.3.2.1 Types of distributional shifts

Monitoring of AI models is achieved by assessing shifts in three primary distributions associated with AI models. These distributional shifts are:

a) **Covariate shift** refers to a shift in distribution of the input features between the source and target domains while the input/output (the model) relationship remains unchanged [63],

EXAMPLE 1 Target domain has frequent occurrence of low-light driving conditions, but source (training) data did not include sufficient data from low light conditions. The deployed ML model will make confident but incorrect predictions in the now predominantly low light driving conditions.

b) **Label shift,** also known as prior shift, refers to a shift in the distribution of the model's output. Label shift arises when class proportions of the labels differ between the source and target, but the input distributions of each class and the input/output (the model) relationship remain the same. One pathological case is target data imbalance between source and target domains.

EXAMPLE 2 Training was performed on clearly separated classes of humans and vehicles, but the target domain is a very crowded scene of multiple variants of vehicles and people. The deployed ML model might totally ignore certain vehicle classes.

c) **Model shift or concept shift** is a shift in the input/output relationship describing the model. That is "same input different output". Concept or model shift can occur if the environment itself shifts between training and deployment. Concept shift can be gradual, sudden or recurrent.

EXAMPLE 3 Training data was from driving environments that do not allow turns at red lights, but the target domain environment allows turns on red light. The deployed ML system cannot anticipate road users will turn on red lights.

G.3.2.2 Distributional shift monitoring

This clause discusses monitoring of AI models post deployment for detecting and quantifying shifts in the behaviour of AI models. The clause also describes the need for mitigating actions to manage the impacts from such distributional shifts on the performance of deployed AI models.

NOTE 1 For the purpose of this document, shift implies distributional shift.

NOTE 2 A deployed AI model is said to have shifted (or drifted) if the response of the AI model in deployment differs from the expected response achieved during development.

The development environment is referred to as "Source".

The deployment environment is referred to as "Target".

NOTE 3 Shift in the behaviour of a deployed AI model will occur when the distributions of data and/or model in the target domain are different from those in the source domain.

NOTE 4 Model monitoring methods are essential for monitoring the behaviour of AI models in the target environment.

© ISO 2024 – All rights reserved

There are three fundamental objectives of any monitoring strategy:

— Detect: in the target domain, detect the occurrence of one or more of the three primary distributional shifts, preferably, with as few samples as possible.

— Quantify: characterize and quantify the distributional shifts associated with the deployed AI model.

— Mitigate: set of architectural and development measures that mitigate the impact of shifts.

G.3.2.3 Distributional shift detection and mitigation

The AI system can provide alerts when facing unexpected inputs, and/or changes in the input distribution. ML models can also provide a confidence score associated with their predictions. There are typically two classes of approaches that can be adopted for monitoring shifts and providing alerts.

a) Distribution based methods rely on estimating distance between distributions such as KL-divergence, or population stability index [64].

b) Anomaly detection-based methods rely on single point out of distribution methods such as anomaly or outlier detection.

While detecting data shift is critical, actions that mitigate the impact of shifts also are considered. Where possible, shift-related risk prevention actions are considered during development. Additionally, shift-related risk reduction provisions are provided during deployment, these include offline methods and online methods:

c) Architectural measures: Any measures that reduce model complexity and tendency to overfit. Feature ablation is one approach that eliminates features that are sensitive to shift but have small predictive power.

d) Development measures: Development measures are data adequacy measures that reduce the incidence/severity of distributional shifts post deployment. Methods include data diversity with adequate and balanced representation of the operating domain, including but not limited to variations in sensing (sensor state) and environment [65]. Data augmentation via synthetic data sources is another approach for achieving data sufficiency. However synthetic data might not be the primary data source for model development. AI safety requirements related to data to minimize the incidence of shifts in AI models post deployment are covered in Clause 11.

NOTE 1 Development phase mitigation approaches such as importance-weighted empirical risk minimization (IWERM) can be used but this method relies on prior knowledge of the target domain [66].

e) Post-deployment measures: Offline methods refer to techniques where shifts are determined from in-use data that has been off-boarded to a cloud server.

NOTE 2 Data shifts are identified and quantified. If the effect of distributional shift is proven or estimated to be leading to contributing AI error, the associated model/s are retrained in an offline manner and the new weight/biases are updated to all impacted assets. This is a type of iterative learning. Domain randomization is another offline technique leveraged during training and relies on exhaustive simulations.

Online methods refer to techniques where any retraining or adaptation is performed in real-time.

NOTE 3 One online approach often utilized is to have an ensemble of inferencing models, where a comparison between the predictions of the main and surrogate models are utilized to provide prediction uncertainty. Online methods can be subject to other requirements.

G.4 Training safety measures

G.4.1 Hyperparameter tuning

© ISO 2024 – All rights reserved

Hyperparameters are model-training related parameters that cannot be learned from data and are set prior to training. The choice of hyperparameters can have a significant impact on model performance and its robustness and hence a robust set of hyperparameters must be determined.

Typical hyperparameters are for DNNs: learning rate, number of hidden layers, number of units per layer, batch size, number of training epochs, regularization parameters, activation functions, dropout rates etc.

Typical methods used for hyperparameter tuning include: random search (large set of hyperparameters), grid search (small set of hyperparameters), Bayesian optimization, hyperparameter scaling, etc. Batch normalization is another approach that can make an AI model robust to the choice of hyperparameters.

In practice hyperparameter tuning can be done either at a single model level though systematic iterations or through running multiple models simultaneously.

Hyperparameter tuning allows fine-tuning a model's to achieve the best balance between underfitting and overfitting, leading to better overall performance. Underfitting can be managed by increasing model complexity via tuning hyperparameters like adding more layers or units can help the model capture more intricate patterns in the data. Overfitting can be addressed by reducing model complexity through regularization or adjusting learning rates can prevent the model from fitting the training data too closely and improve its generalization to unseen data.

G.4.2 Robust Learning

Robustness of AI systems is very relevant for achieving safety of the intended performance. Good robustness is in many ways an extension of good generalization of an AI system. Therefore, a robust AI system is one that is invariant to small perturbations in the inputs. Typical sources of input perturbations include adversarial input examples and noise corrupted input examples. Methods for improving robustness include both training and architectural methods. Some typical methods for improving the robustness of AI models are discussed.

1) **Data pre-processing** techniques such as cleaning, normalization and scaling are training time methods that improve model robustness.

2) **Regularization** is a training approach to prevent AI models from over-fitting. This improves a model's generalizability and stability making the model robust to unseen or new data. Typical regularization methods include L_1, L_2, Lasso, Dropout etc.

3) **Adversarial training** is a learning method that improves model robustness by injecting noise into the input during training, thereby mimicking attacks. One such approach to improve adversarial robustness is to use randomized smoothing.

4) **Error analysis methods** help isolate patterns in a model's behaviour and can be used to inform model adjustments for improved robustness.

5) **Ensemble methods** such as bagging, boosting and stacking help improve robustness by combining predictions from multiple models, however, these are computationally expensive methods and might not be preferred for compute constrained applications. Please refer to G.1.1.2 for details.

6) **Domain Randomization (or generalization) approache**s make the models robust to domain shifts, where the source and target domains are shifted (sunlight to low light) but the tasks remain the same. This can also be considered as a data augmentation technique. Domain randomization also makes the models more robust to real world scenarios.

7) **Fault Aware Training**: When the AI system disturbances are predictable (e.g. for hardware errors), fault-aware training that includes error modelling during training [67] can be used to make neural networks resilient to specific fault models on the device.

G.4.3 Transfer learning

© ISO 2024 – All rights reserved

Transfer learning is an effective method to leverage some knowledge learned from a possibly different source domain and tasks and reuse it for the target domain and tasks. Transfer learning can help alleviate a lack of target domain data or simply speed up the task(s) performance increase [68]. Transfer learning methods can be classified based:

— on the availability of information for the source and target domains; methods are classified as "Transductive transfer learning", "Inductive transfer learning" or "Unsupervised transfer learning".

— on the level of consistency between the sources and target features and label spaces; methods are classified as "Homogeneous transfer learning" or "Heterogeneous transfer learning".

In re-using a model trained on a large corpus of data, transfer learning can foster generalization and robustness of the model resulting from the transfer [69]. However, transfer learning presents some challenges:

— Risk of emergent properties, such as biases, inherited from the source model.

— Transfer learning can result in reduction of performance compared with a model training solely on the source domain without transfer learning. This is referred to as "negative transfer" and might occur when there is not enough relatedness between the source and target domain or tasks, or between the source and target feature spaces.

An appropriate assessment of relatedness between the source and target domains, tasks and features, followed by an appropriate extraction of the knowledge to transfer, can justify the use of transfer learning for safety-related AI systems. Sufficient V&V to meet the target domain KPIs is also required to ensure the effectiveness of the transfer learning.

NOTE If the source model is complex then the explainability and transparency of the model resulting from transfer learning can be adversely affected.

G.4.4 Confidence calibration and uncertainty quantification of AI models

The prediction certainty of the AI components can be estimated, where applicable, and calibrated for high certainty of predicted confidence during design time. The confidence score of a predictor is defined as the predicted probability of correctness, ex. 90% confident this is a car. The observed accuracy, however, is the observed (in-use) probability of correctness, ex. 8 correct predictions on 10 predictions is an 80% observed accuracy. Using these metrics, a calibration error (CE) can be calculated as: Calibration Error = |Confidence Score – Observed Accuracy|. For AI models that do not explicitly report a confidence score, the predictive certainty can still be established at design time via measures that establish desired statistical properties of predictions over expected variations of the covariates that cover anticipated noise factors in real life applications. These can include simple input perturbations such as, for example, variations in lighting conditions. Typically, the calibration error is minimized during design time. There are several techniques commonly used to calibrate model's predictive certainty. These include, Brier Score, Temperature Scaling, Isotonic Regression, Platt Scaling, etc. It is important that a suitable scoring rule be used to encourage honest forecasts by the predictor.

A model's predictive certainty and calibration error is re-evaluated periodically during a models life-cycle. This is especially critical for any retraining of a model.

G.4.5 Verifying feature selection

Verification of feature importance or relevance can support the development of AI models that fulfil their AI safety requirements. An optimal feature set not only prevents overfitting and improves generalizability but also makes a model more interpretable. Feature verification is a combination of data pre-processing, feature selection/engineering, model training and testing, feature importance and feature sensitivity analyses[70]. For deep learning models, such as computer vision models, feature selection can be automated such as through the use of convolution filter kernels (edge detectors). However, features can still be studied via visualization

© ISO 2024 – All rights reserved

of feature heat maps at various layers of the model. Saliency maps are an example of such a method and are used for understanding visual properties of an image to help improve performance of computer vision tasks. Another approach is to use Shapley values that attribute to each feature their contribution to the final output.

G.4.6 Monitoring multiple scores

Monitoring multiple scores during model training provides a more comprehensive picture of a model's performance and its insufficiencies. Additionally monitoring multiple scores also helps identify and reduce overfitting and thereby improves a models generalization capability. There are several well-known model scoring metrics that can be used to choose the appropriate scoring method for the AI model being used. For example, accuracy is a common metric used to evaluate classification models, but it might not be the best metric to use if the classes are imbalanced. In such cases, metrics such as precision, recall, and F1-score might provide more useful information about the model's performance. Some commonly used metrics include:

— Accuracy

— Precision

— Recall

— F1-Score

— Area under the Receiver Operator Characteristic Curve

— Mean average precision

— Intersection over union

G.4.7 Attention or saliency Maps

Attention or saliency maps can improve the explainablity of AI components by providing explanations/ visual aids why the AI component made certain prediction, which features led the AI component to take certain decisions [71] or which regions of the input are more important for prediction of AI component [72].

The explanations provided by the attention maps can be reviewed by the stakeholders for identifying the problems in the AI component architecture or development process (e.g. training set improvement) such that they can enhance or introduce other architectural or development measures.

NOTE The properties of the explanation such as completeness and correctness are heavily dependent on aspects such as the quality and the quantity of the inputs considered during the training process.

Using the visualization provided by attention and saliency maps can improve the robustness [73] and reduce the bias of the AI components by focusing more on and the most relevant features of the input space. AI components that can produce more interpretable saliency maps can be more robust to the adversarial inputs [73].

G.4.8 Interpretable latent features

Learned or identified latent features in an AI model can be utilized as evidence to ensure that specific requirements are fulfilled. The methods Supervised concept extraction [74], Unsupervised concept mining [75] and; Concept bottleneck architectures [76] can be used to obtain evidence for learned features, which can be utilized so that an expert can derive possible failures that a learned AI model has, guiding training or update strategies.

G.4.9 Augmentation of Data

Data augmentation techniques can improve the performance and robustness of machine learning models by increasing the amount and diversity of training data, reducing sensitivity to noise, and improving

© ISO 2024 – All rights reserved

generalization to unseen data. Data augmentation is useful in a variety of scenarios especially when datasets are small or imbalanced, the model is complex and prone to overfitting. Data augmentation can be applied at different stages during the design phase (train, dev-test) or after evaluating in-use performance. There are additional considerations that inform the need for data augmentation, these include (but are not limited to) an assessment of structural coverage (see G.4.9.1) as well as test data coverage (see G.4.9.2).

G.4.9.1 Structural coverage of an AI component

Some methods for model coverages are proposed and are based on Structural coverage metrics at the software unit level in ISO 26262. In these methods, augmentation or filtering of input data based on structural Coverage of AI component is used:

1) to ensure comprehensiveness of testing,

2) to identify an undesired behaviour of the AI model, and

3) to obtain methodical evidence towards robustness

Examples of coverage methods are:

— Neuron Coverage: assessing if all neurons are activated during testing [77];

— Sign-Sign Coverage: confirming that a change of sign in one or more AI component inputs has the desired change of sign in one more AI component outputs [78];

— Sign-Value Coverage: confirming that a change of sign in one or more AI component inputs has the desired change of value in one more AI component outputs;

— Value-Value Coverage: confirming that a change of value in one or more AI component; inputs has the desired change of value in one more AI component outputs;

— Value-Sign Coverage: confirming that a change of value in one or more AI component inputs has the desired change of sign in one more AI component outputs.

NOTE As these methods are white or open box methods, they can help improve the overall trustworthiness in the AI components by building confidence due to some level of transparency and possibly explainability. However, to be adopted as state-of the art, these methods need to be developed further to have more solid and evidence-based relationship with the AI properties.

G.4.9.2 Data coverage techniques for test data augmentation

Data coverage techniques based on dataset characteristics[65] can be used to inform data augmentation. A suitable method is used to determine the appropriate augmentation method(s).

EXAMPLE Data augmentation can be informed via data coverage techniques such as:

1) equivalence partitioning: partitions a set of test cases into groups with each group representing a different scenario. The objective is to test each partition with an equivalent set of test cases.

2) centroid positioning: this is a technique for selecting test cases that are representative of the mean of the data distribution and represents an “on average” performance of the model.

3) boundary conditioning: this technique selects test case at the boundaries/edge of the distribution representing the input domain.

4) pair-wise boundary conditioning: in this method pairs of boundary test cases are selected to cover all combinations of edge cases.

© ISO 2024 – All rights reserved

The method helps identifying the potential bias in the data (e.g. by using equivalence partitioning). By augmenting the dataset based on the defined techniques this method can also help identify issues in the AI model related to accuracy and thereby supporting predictability improvement. This method can also help identify the feature space relevance for prediction by understanding sensitive dataset properties there by supporting transparency and interpretability.

G.5 Monitoring and AI system modification

G.5.1 Dynamic Environment Monitoring

The AI system and AI components are designed by making certain assumptions about the environment that they need to operate in. During the design time their behaviour is verified and validated to ensure that they operate correctly when all the assumption are met. However when certain assumptions become invalid it is likely that the correct functioning of AI system / AI components cannot be guaranteed. Dynamic environment monitoring involves monitoring assumptions made during design at runtime such that if they become invalid, appropriate actions can be taken to ensure continued AI safety [57].

EXAMPLE The example assumption can include the relative speed of other traffic participants, weather conditions, lighting conditions, road conditions (e.g., construction zone).

The capabilities for realising this measure can include specific AI components for monitoring the input space and operating environment. Such components can be non-AI based components or less complex AI components as they will be acting as a safety measure. Where necessary to detect the violation of the assumptions, appropriate threshold for the attributes of the operating environment can be considered.

G.5.2 AI Model Modification

AI models can become stale or lose performance over time due to a variety of reasons, such as from distributional shift. These include design time issues such as inherently inadequate predictive power of the model, data and/or algorithmic bias and deployment (or in use) issues such as shift in distributions between training and deployment and the presence of outliers or out-of-distribution (OOD) examples.

To overcome such problems AI models might need to be updated/re-trained. However, retraining or continual learning of AI models can be challenging and issues such as catastrophic forgetting[79][80], catastrophic remembering [81] etc. need to be addressed where necessary.

G.5.2.1 Criteria for Retraining

The quality of the safety related performance of an AI element is defined by its input space and the functionalities (behavioural, prediction, etc.) required to fulfil the intended mission within the operational domain (OD) i.e., the environment where the system is deployed.

Partial or full retraining is performed when there is evidence that the AI system can no longer safely fulfil its mission. These inadequacies of the AI system might result from:

— A mismatch between the AI systems' input space and its OD.

— Shifts in distributions between training and deployment (see section Clause G.3), for example the level of traffic density increased over time, there are new agents to interact with, there are new road signs etc.

— A domain shift where the AI system is deployed in an OD that is not within a reasonable generalization distance from the input space used for development. For example, AV driving policies developed for country A are deployed in country B.

— A mismatch between the learnt competencies of an AI system and the competencies required in deployment. This can result from:

© ISO 2024 – All rights reserved

a) The proven inability or insufficient ability of the AI system to manage situations within its input space, for example incidents, near-misses or misdiagnoses are reported.

b) A change in the regulations or in the relevant safety standards covering the input space so that expected performance levels are modified, for example the tolerated rate of false negative is reduced.

NOTE Partial or full retraining cannot be dictated by the type of discrepancy or the amplitude of the discrepancy but by the confidence in either approach to guarantee that the discrepancy is addressed and no safety-critical regression results from the retraining.

G.5.2.2 Targeted and controlled model update

Successive iterations of developing and training an AI system can be required in order to achieve the AI safety requirements.Retraining with additional data might completely change the behaviour of model, and thus the controllability of the outcome is limited, i.e., difficult to address target insufficiencies or prone to introduce undesirable regression.

For traditional program code, we have clear modularization and thus can analyse which functions are likely to be affected when changes are made, and which tests are to be executed again. In the case of ML models, all the tests are executed again as change in the outcome might not be able to systematically predicted. However, in some cases, differential testing can be used to understand the differences between the original and re-trained models. For example, the inputs where the current model fails but the previous version succeeded, are examined.

There have been the following approaches to address this problem by targeted and controlled model update:

— Additional optimisation objectives (e.g., in the loss function) can be used to penalise regressions, i.e., failures on inputs for which a previous version of the model succeeded.The balance between improvement and regression can be investigated as a hyperparameter such as weights between the original fitness and the penalty.

— Model repair techniques identify neuronsor parameters suspicious/responsible for critical errors or significant successes. Then, important parameters for successful behaviour in the past version can be frozen or focus on re-optimization of the undesirable behaviour in the current version. This way of model update can be done directly by focused optimization or indirectly by data augmentation that stimulate the relevant neurons.

NOTE Retraining can be implemented in a manner that avoids model regression. Regression of model performance can be established based on the model's training history [79].

G.6 Alignment of intention

AI Alignment addresses the concern of AI system's behaviours in achieving some human-designed objectives being not aligned with human-level expectations with respect to values and objectives. Complex AI systems might seek to aggressively achieve an objective with disregard for other factors that can result in harmful, unsafe or unethical behaviours. Typically misalignment is a result of a mismatch between human-defined objectives and values and the behaviour the AI system exhibits in achieving those objectives. As an example, a shortest-time requirement on a self-driving car is by itself an incomplete requirement and might lead to the AI system acting in a hazardous manner to achieve this objective. The same requirement subject to constraints that prevent violation of safety or other values can be designed to achieve full alignment. For example, Qualitative methods include Value specification, Adversarial Testing, or Ethical Framework and Quantitative measures could include, Reward Modeling, Robustness testing, etc.

G.7 Considerations related to the target execution environment

© ISO 2024 – All rights reserved

This clause relates to requirement 10.3.11 regarding the demonstration of AI safety within the target execution environment.Classes of evidence that support this claim include targeted safety analyses and assumptions on the execution integrity of the target platform.

G.7.1 Optimization of parameters and optimization of architectural entities of AI components

Optimizations of parameters and model architectures can be applied to meet the constraints of the target execution environment. The optimizations can alter the size and the complexity of the AI components such that lesser spatial and temporal resources are required. The basis datatype used for the target execution environment can be smaller in size, precision and accuracy than the one that is used during training. Also, certain neurons, their weights, their filters, or channel can be pruned. In addition to fulfil the target constraints, pruning can lead to improve the computation efficiency of AI components.

Pruning can lead to better generalization by preventing overfitting. The reduced network dimension can contribute towards the improving the interpretability using visualization methods, analysability of AI component architecture.

G.7.2 Knowledge distillation also known as teacher-student model

Optimized models can be derived from larger or more complex models to meet target execution environment constraints. Hence, optimized AI-models are trained to:

— follow larger model behaviour;

— minimize differences between the outputs of both models.

Knowledge distillation can be applied to improve generalization and accuracy of optimized AI-models compared to the larger AI-model they are derived from. Since optimized AI-models are usually lower in complexity and size the risk related to interpretability will also be reduced.

G.7.3 Analysis for differences

There are many factors that can be different between the development environment and target execution environment such as:

— Hardware (e.g., GPU, accelerators) and Hardware Resources (e.g., memory, storage);

— Supported data size;

— Software Dependencies (e.g., libraries, operating system);

— Tools (e.g., compiler).

Such an argument is based on thorough analysis of all differences. However, in case detailed analysis of AI-model differences is not feasible (e.g., due to the high complexity of AI-models) specific acceptance tests can be run on both development and target execution environment. Test results can be compared to identify any potential increase of risk and build or support the argument.

© ISO 2024 – All rights reserved

Annex H

Typical performance metrics for machine learning

In this annex, some of the widely-adopted performance metrics for machine learning is described. Here, these metrics are categorised into metrics for regression and metrics for classification.

a) **Metrics for regression**

© ISO 2024 – All rights reserved

— **Mean Squared Error (MSE):** The average of squared difference between the ground truth values and the predicted values from the model.

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2, \qquad \text{Formula H–1}$$

where Y is ground truth, $\hat{Y}$ is the predicted output, and N is the number of datums.

Range: $[0, \infty]$, with 0 being the best.

MSE is a differentiable metric and can be well optimized. However, MSE penalizes small errors (by squaring the terms), leading essentially to an over-estimation of how bad the model is.

— **Mean Absolute Error (MAE):** The average of the absolute differences between the ground truth values and predicted values from the model.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} \left|Y_i - \hat{Y}_i\right|, \qquad \text{Formula H–2}$$

where Y is ground truth, $\hat{Y}$ is the predicted output, and N is the number of datums.

Range: $[0, \infty]$, with 0 being the best.

MAE is similar to MSE but more robust towards outliers than MSE, as it does not exaggerate errors by squaring those terms. MAE indicates how far the predictions were from the ground truth. However, MAE does not indicate the direction of the error, i.e. whether the data were under-predicted or over-predicted. MAE is typically used to assess how close the predictions are to the ground truth on average.

— **Mean Absolute Percentage Error (MAPE):** Average absolute percentage difference between predicted values and actual values.

$$MAPE = \frac{1}{N}\sum_{i=1}^{N} \frac{|(A_i - F_i)|}{A_i}, \qquad \text{Formula H–3}$$

where N is the number of fitted points, A_i is the actual value, and F_i is the predicted value.

Range: [0,1], with 0 being the best.

MAPE measures the average magnitude of error produced by a model, or how far off predictions are on average. It is often used as the loss function in regression problems and forecasting models due to the intuitive interpretation in terms of relative error for evaluation. MAPE is not suggested to be used when actual values can be at or close to zero.

— **Root Mean Squared Error (RMSE):** Square root of the average of the squared difference between the ground truth values and the predicted values from the model.

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2} \qquad \text{Formula H–4}$$

where Y is ground truth, $\hat{Y}$ is the predicted output, and N is the number of datums.

© ISO 2024 – All rights reserved

Range: $[0, \infty]$, with 0 being the best.

RMSE has an advantage over MSE with the handling of the penalization of smaller error by square rooting the error terms. RMSE is often used for large numbers (prediction or ground truth) for hyper-parameter tuning or batch training a ML model. RMSE focuses on penalizing large errors.

— **R-Squared:** a measure of the proportion of the variance of a dependent variable that is explained by the regression model.

$$R^2 = 1 - \frac{\sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2}{\sum_{i=1}^{N} (Y_i - \overline{Y}_i)^2}$$ Formula H–5

where Y is ground truth, $\hat{Y}$ is the predicted output, $\overline{Y}$ is the mean of dependent variables,

and N is the number of datums.

Range: $[-\infty, 1]$, with 1 being the best. Negative score of R-Squared indicates that the regression model is erroneous. The lower the error in the regression analysis relative to total error, the higher the R-squared value will be. Any R-squared value greater than zero means that the regression analysis did better than just using a horizontal line through the mean value. In the rare cases that R- squared value is negative, the regression analysis need to be re-evaluated, especially if an intercept is forced.

R-Squared is a relative metric used to compare the model with other models trained on the same dataset. R-Squared indicates the difference between samples in the dataset and the predictions made by the model.

b) **Metrics for classification**

© ISO 2024 – All rights reserved

For classification problems, the results are grouped into four categories

— True Positive (TP), when both the predicted values and ground truth values are 1;

— True Negative (TN), when both the predicted values and ground truth values are 0;

— False Positive (FP), when the ground truth value is 0 but the predicted value is 1;

— False Negative (FN), when the ground truth value is 1 but the predicted value is 0;

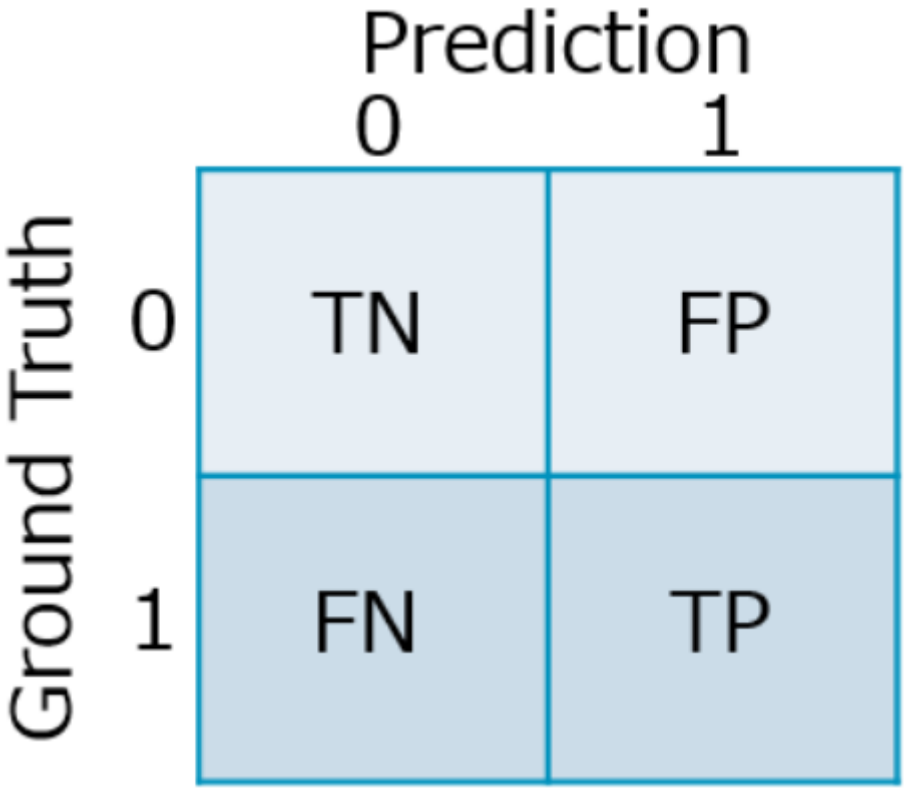
— Accuracy [82]: The ratio of the number of correct predictions and the total number of predictions.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$ Formula H–6

Range: [0,1], with 1 being the best.

Accuracy is the simplest metric that is often used for the classification problems which are well balanced. However, Accuracy alone does not serve as a reliable metric with class-imbalanced dataset, where a significant disparity between the number of positive and negative labels exists.

— **Confusion Matrix [83]:** A table to visualize the performance of an algorithm (see Figure H–1), typically, a supervised learning one, on a set of the test data for which the ground truth values are known.



© ISO 2024 – All rights reserved

Figure H–1 — Table of confusion matrix

Confusion Matrix is an intuitive and descriptive metrics used to find the accuracy and correctness of a machine learning algorithm. It is mainly used where the output can contain two or more types of classes.

— **Precision and Recall [82]:** Precision for a label is defined as the number of true positives divided by the number of predicted positives. Recall for a label is defined as the number of the true positives divided by the total number of actual positives.

$$Precision = \frac{TP}{TP+FP}$$ Formula H–7

$$Recall = \frac{TP}{TP+FN}$$ Formula H–8

Precision range: [0,1], with 1 being the best.

Recall range: [0,1], with 1 being the best.

In contrast to Accuracy, Precision and Recall are two important metrics for performance evaluation from different aspects for imbalanced dataset. Models inherently trade off Precision and Recall, and they are used differently based on specific use case requirements. Precision is often used when false negative is less emphasized, while Recall is often preferred for output-sensitive predictions.

Mean Average Precision: The average of Average Precision (weighted mean of precisions at each threshold) of each class.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i$$ Formula H–9

where AP_i is the Average Precision of class , and is the number of classes.

Range: [0,1], with 1 being the best.

mAP incorporates the trade-off between Precision and Recall, and considers both False Positive and False Negative. mAP is a suitable metric for most detection applications.

— **F1-score [82]:** The harmonic mean of Precision and Recall

$$F1 - score = 2\frac{Precision \times Recall}{Precision + Recall}$$ Formula H–10

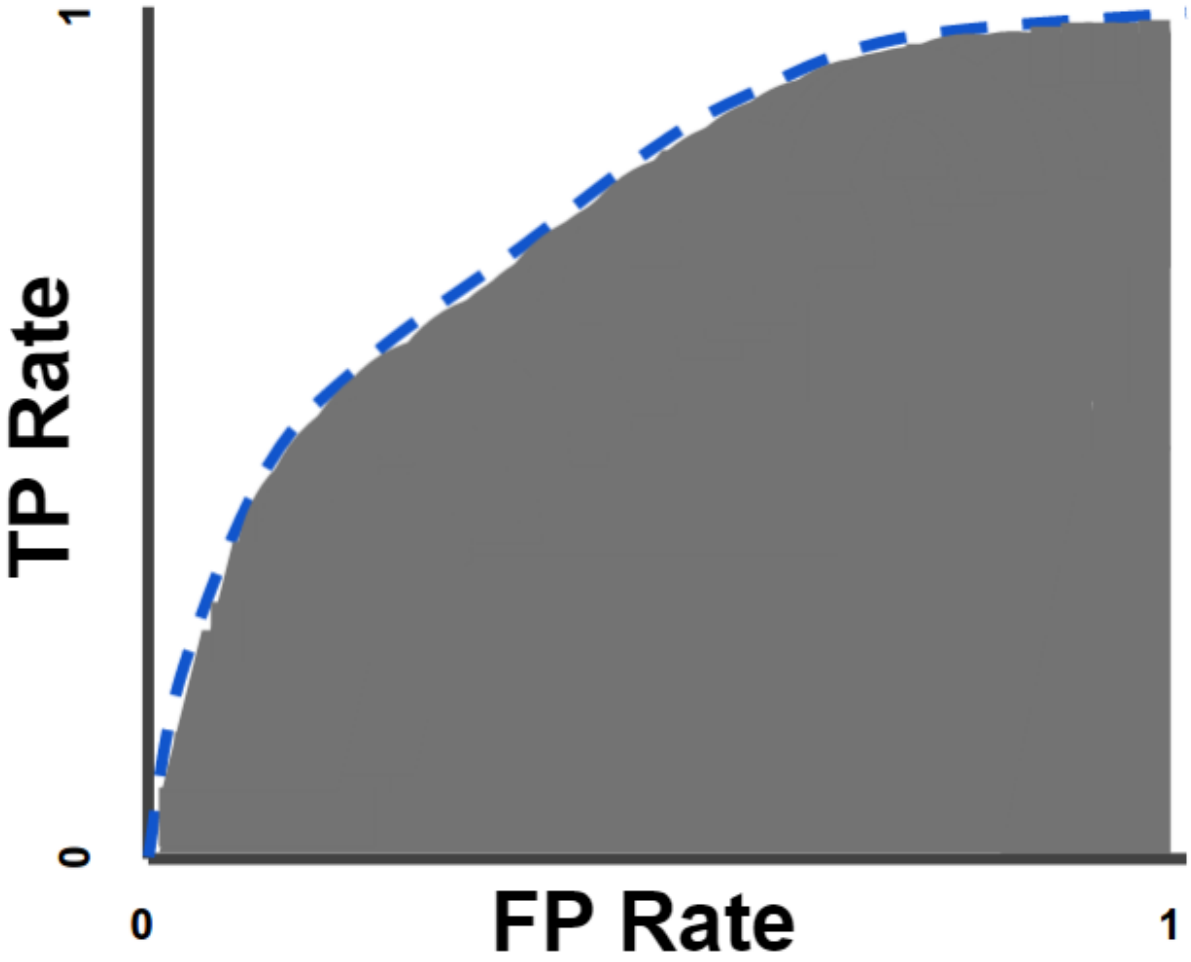
Range: [0,1].

F1-score, also known as f-score or f-measure, is used to measure a test's accuracy. It indicates how precise the classifier is (i.e., how many instances it classifies correctly), as well as how robust it is (i.e., does not miss a significant number of instances).

F1-score maintains a balance between Precision and Recall for the classifier. However, F1-score gives equal weight to precision and recall.

© ISO 2024 – All rights reserved

— **AU-ROC (Area under Receiver Operating Characteristics Curve) [82]:** an ROC curve is a graph showing the performance of a classification model at all classification thresholds, with two parameters True Positive Rate (TPR) and False Positive Rate (FPR), which are defined as $TPR = \frac{TP}{TP+FN}$, and $FPR = \frac{FP}{FP+TN}$. AU-ROC measures the entire two-dimensional area underneath the entire ROC curve as shown in Figure H–2.



© ISO 2024 – All rights reserved

Figure H–2 — ROC curve and AU-ROC

Range: [0,1].

AU-ROC provides an aggregate measure of performance across all possible classification thresholds. It is particularly useful if the importance of positive and negative classes is equal for evaluation purpose. One way of interpreting AUC is as the probability that the model ranks a random positive example higher than a random negative example. AU-ROC is scale-invariant, it measures how well the predictions are ranked rather than their absolute values. AU-ROC is also classification-threshold-invariant, meaning that it measures the quality of the model's prediction regardless of what classification threshold is selected.

NOTE 1 In real applications, using a single metric mentioned above could result in biases in system performance and are not adequate to evaluate the system performance within a certain level of confidence. A combination of multiple metrics from different angles could lead to more reliable, balanced and comprehensive conclusion of the system performance.

NOTE 2 An AI system does not render predictions/decisions on a general scope. Instead, the AI system focuses on particular use cases. A specific use case demands careful consideration, choice, and adaptation of metrics. Same score for the same metric does not necessarily indicate the same level of performance for a different application.

© ISO 2024 – All rights reserved

Bibliography

[1] ISO 21448:2022, *Road vehicles — Safety of the intended functionality*

[2] ISO/SAE PAS 22736:2021, *Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles*

[3] ISO/IEC Guide 51:2014, *Safety aspects — Guidelines for their inclusion in standards*

[4] ISO/IEC TR 5469, *Artificial intelligence — Functional safety and AI systems*

[6] Goal Structuring Notation Community Standard, Version 3, The Assurance Case Working Group (ACWG), SCSC-141C, https://scsc.uk/gsn?page=gsn%202standard

[7] Claims Evidence Argument (CAE) Framework, https://claimsargumentsevidence.org/

[8] Structured Assurance Case Metamodel (SACM), v2.2, The Object Management Group, 2021, https://www.omg.org/spec/SACM.

[9] ISO 26262-5:2018, *Road vehicles — Functional safety — Part 5: Product development at the hardware level*

[11] Hawkins, R., Kelly, T., Knight, J. and Graydon, P., 2011. A new approach to creating clear safety arguments. In Advances in systems safety (pp. 3-23). Springer, London.

[12] IEC 61508-4:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 4: Definitions and abbreviations (see Functional Safety and IEC 61508)*

[13] Mood, A. M.; Graybill, F. A.; Boes, D. C. (1974). "Section 2.3". Introduction to the Theory of Statistics (3rd ed.). McGraw-Hill. ISBN 0070428646

[14] Czarnecki, Krzysztof, and Rick Salay. "Towards a framework to manage perceptual uncertainty for safe automated driving." International Conference on Computer Safety, Reliability, and Security. Springer, Cham, 2018.

[15] Montavon, G., Binder, A., Lapuschkin, S., Samek, W., Müller, KR. (2019). Layer-Wise Relevance Propagation: An Overview. In: Samek, W., Montavon, G., Vedaldi, A., Hansen, L., Müller, KR. (eds) Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. Lecture Notes in Computer Science, vol 11700. Springer, Cham. https://doi.org/10.1007/978-3-030-28954-6_10

[16] Delseny, Hervé, et al. "White paper machine learning in certified systems." arXiv preprint arXiv:2103.10529 (2021).

[17] Willers, Oliver, et al. "Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks." International Conference on Computer Safety, Reliability, and Security. Springer, Cham, 2020.

[18] Vapnik, Vladimir N. 2000. The Nature of Statistical Learning Theory. Information Science and Statistics. Springer-Verlag.

[19] Hippenstiel R. D., Detection theory: Applications and Digital Signal Processing, 2002.

© ISO 2024 – All rights reserved

[20] Jie M. Zhang, Mark Harman, Lei Ma, Yang Liu, “Machine Learning Testing: Survey, Landscapes and Horizons”, IEEE Transactions on Software Engineering (Volume: 48, Issue: 1, 01 January 2022)

[21] Wassim G. Najm, John D. Smith, Mikio Yanagisawa: Pre-Crash Scenario Typology for Crash Avoidance Research. Proceeding of the 20th international technical conference on the enhanced safety of vehicles 2007.

[22] Oliver Zendel, Katrin Honauer, Markus Murschitz, Daniel Steininger, Fernandez Gustavo Dominguez : WildDash - Creating Hazard-Aware Benchmarks. ECCV 2018.

[23] Zhou, Zhi Quan, and Liqun Sun. "Metamorphic testing of driverless cars." Communications of the ACM 62.3 (2019): 61-67.

[24] Zhang, Mengshi, et al. "DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems." 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE). IEEE, 2018.

[25] Nie, Changhai, and Hareton Leung. "A survey of combinatorial testing." ACM Computing Surveys (CSUR) 43.2 (2011): 1-29.

[26] Cheng, Chih-Hong, Chung-Hao Huang, and Georg Nührenberg. "nn-dependability-kit: Engineering neural networks for safety-critical autonomous driving systems." 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). IEEE, 2019.

[27] Gladisch, Christoph, et al. "Leveraging combinatorial testing for safety-critical computer vision datasets." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. 2020.

[28] Szegedy, Christian, et al. "Intriguing properties of neural networks." arXiv preprint arXiv:1312.6199 (2013).

[29] Kurakin, Alexey, Ian J. Goodfellow, and Samy Bengio. "Adversarial examples in the physical world." Artificial intelligence safety and security. Chapman and Hall/CRC, 2018. 99-112.

[30] Abbas, Houssam, and Georgios Fainekos. "Convergence proofs for simulated annealing falsification of safety properties." 2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton). IEEE, 2012.

[31] Deshmukh, Jyotirmoy, et al. "Testing cyber-physical systems through Bayesian optimization." ACM Transactions on Embedded Computing Systems (TECS) 16.5s (2017): 1-18.

[32] Pietrantuono, Roberto, and Stefano Russo. "Probabilistic sampling-based testing for accelerated reliability assessment." 2018 IEEE International Conference on Software Quality, Reliability and Security (QRS). IEEE, 2018.

[33] Katz, Guy, et al. "Reluplex: An efficient SMT solver for verifying deep neural networks." International conference on computer aided verification. Springer, Cham, 2017.

[34] Gehr, Timon, et al. "Ai2: Safety and robustness certification of neural networks with abstract interpretation." 2018 IEEE symposium on security and privacy (SP). IEEE, 2018.

[35] Tran, Hoang-Dung, et al. "Verification of deep convolutional neural networks using imagestars." International conference on computer aided verification. Springer, Cham, 2020.

© ISO 2024 – All rights reserved

[36] Cheng, Chih-Hong, et al. "Towards safety verification of direct perception neural networks." 2020 Design, Automation & Test in Europe Conference & Exhibition (DATE). IEEE, 2020.

[37] Sinan Hasirlioglu, A Novel Method for Simulation-based Testing and Validation of Automotive Surround Sensors under Adverse Weather Conditions, Doctoral Thesis, Institute for Pervasive Computing, Johannes Kepler University Linz, 2020.

[38] Hejase, Mohammad, et al. "A Validation Methodology for the Minimization of Unknown Unknowns in Autonomous Vehicle Systems." 2020 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2020.

[39] Li, Changwen, et al. "ComOpT: Combination and Optimization for Testing Autonomous Driving Systems." 2022 International Conference on Robotics and Automation (ICRA). IEEE, 2022.

[40] ISO 3534-1:2006, *Statistics — Vocabulary and symbols — Part 1: General statistical terms and terms used in probability*

[41] Ali, Nazakat, Manzoor Hussain, and Jang-Eui Hong. 2020. “Analyzing safety of collaborative cyber-physical systems considering variability.” IEEE Access (IEEE) 8: 162701–162713.

[42] Kramer, Birte, Christian Neurohr, Matthias Büker, Eckard Böde, Martin Fränzle, and Werner Damm. 2020. “Identification and quantification of hazardous scenarios for automated driving.” Model-Based Safety and Assessment: 7th International Symposium, IMBSA 2020, Lisbon, Portugal, September 14–16, 2020, Proceedings 7. 163–178.

[43] Adee, Ahmad, Roman Gansch, and Peter Liggesmeyer. 2021. “Systematic modeling approach for environmental perception limitations in automated driving.” 2021 17th European Dependable Computing Conference (EDCC). 103–110.

[44] Adee, Ahmad, Roman Gansch, Peter Liggesmeyer, Claudius Glaeser, and Florian Drews. 2021. “Discovery of perception performance limiting triggering conditions in automated driving.” 2021 5th International Conference on System Reliability and Safety (ICSRS). 248–257.

[45] Berk, Mario, Olaf Schubert, Hans-Martin Kroll, Boris Buschardt, and Daniel Straub. 2020. “Assessing the safety of environment perception in automated driving vehicles.” SAE International journal of transportation safety (JSTOR) 8: 49–74.

[46] Salay, Rick, Matt Angus, and Krzysztof Czarnecki. 2019. “A safety analysis method for perceptual components in automated driving.” 2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE). 24–34.

[47] Qi, Yi, Yi Dong, Xingyu Zhao, and Xiaowei Huang. 2023. “STPA for Learning-Enabled Systems: A Survey and A New Method.” arXiv preprint arXiv:2302.10588.

[48] Thomas, Stephen, and Katrina M. Groth. 2021. “Toward a hybrid causal framework for autonomous vehicle safety analysis.” Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability (SAGE Publications Sage UK: London, England) 1748006X211043310.

[49] Qi, Yi, Philippa Ryan Conmy, Wei Huang, Xingyu Zhao, and Xiaowei Huang. 2022. “A hierarchical HAZOP-like safety analysis for learning-enabled systems.” arXiv preprint arXiv:2206.10216.

[50] AIAG and VDA. "EXECUSION OF THE PROCESS FMEA (PFMEA)." In Failure Mode and Effects Analysis - FMEA Handbook: Design FMEA, Process FMEA, Supplemental FMEA for Monitoring & System Response, 79-124. Southfild, MI: Automotive Industry Action Group, 2019.

© ISO 2024 – All rights reserved

[51] Simon, Burton. "A causal model of safety assurance for machine learning" https://arxiv.org/abs/2201.05451

[52] STPA Handbook Nancy G. LEVESON JOHN P. THOMAS MARCH 2018

[53] SAE J3187:2022, *System Theoretic Process Analysis (STPA) Recommended Practices for Evaluations of Automotive Related Safety-Critical Systems*

[54] Simon, Burton, et al. "Safety, Complexity, and Automated Driving: Holistic Perspectives on Safety Assurance." Computer. pp. 22-32, 2021.

[55] Mariusz Bojarski and others, the NVIDIA PilotNet Experiment, arXiv:2010.08776v1 [cs.CV] 17 Oct 2020

[56] Vasu Singh, Mandar Pitale, Impact of Automotive System Safety Design on Machine Learning Based Perception Systems, 2021 4th IEEE International Conference on Industrial Cyber-Physical Systems

[57] Rob Ashmore and others, Assuring the Machine Learning Lifecycle: Desiderata, Methods, and Challenges, arXiv:1905.04223v1

[58] H. Xu, Z. Chen, W. Wu, Z. Jin, S. Kuo and M. Lyu, NV-DNN: Towards Fault-Tolerant DNN Systems with N-Version Programming, 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W), Portland, OR, USA, 2019, pp. 44-47

[59] Fumio Machida, N-version machine learning models for safety critical systems, The DSN Workshop on Dependable and Secure Machine Learning (DSML) 2019

[60] Timo Samann and others, Strategy to Increase the Safety of a DNN-based Perception for HAD Systems, arXiv:2002.08935v1 [cs.CV] 20 Feb 2020

[61] Raul Sena Ferreira and others, Benchmarking Safety Monitors for Image Classifiers with Machine Learning, 26th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2021), IEEE, Dec 2021, Perth, Australia.

[62] Sina Mohseni, Mandar Pitale, JBS Yadawa, Zhangyang Wang; "Self-Supervised Learning for Generalizable Out-of-Distribution Detection", AAAI - 2020

[63] Joaquin Q Candela, et.al, "Dataset shift in machine learning", The MIT press 2009.

[64] Aria Khademi, Michael Hopka, Devesh Upadhyay. 2023. "Model Monitoring and Robustness of In-Use Machine Learning Models: Quantifying Data Distribution Shifts Using Population Stability Index" arXiv preprint arXiv:2302.00775v1

[65] Senthil Mani et. al.; "Coverage Testing of Deep Learning Models using Dataset Characterization", IBM Research, arXiv:1911.07309v1 [cs.LG] 17 Nov 2019

[66] Shimodaira H; "Improving predictive inference under covariate shift by weighting the log-likelihood function"; https://doi.org/10.1016/S0378-3758(00)00115-4

[67] Ussama Zahid et. al; "FAT: Training Neural Networks for Reliable Inference Under Hardware Faults"; arXiv:2011.05873 [cs.LG]

[68] Fuzhen Zhuang et. al.; A Comprehensive Survey on Transfer Learning; arXiv:1911.02685 [cs.LG]

© ISO 2024 – All rights reserved

[69] Rishi Bommasani et. al,; "On the Opportunities and Risks of Foundation Models"; arXiv:2108.07258v3 [cs.LG] 12 Jul 2022

[70] P van de Laar, T Heskes, S Gielen; "Partial retraining: a new approach to input relevance determination"; 1999 Feb;9(1):75-85. doi: 10.1142/s0129065799000071.

[71] Mariusz Bojarski et. al.; "Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car"; NVIDIA Corporation; arXiv:1704.07911v1 [cs.CV] 25 Apr 2017

[72] Vitali Petsiuk et. al.; "Black-box Explanation of Object Detectors via Saliency Maps"; arXiv:2006.03204v2 [cs.CV] 10 Jun 2021

[73] Christian Etmann et. al.; "On the Connection Between Adversarial Robustness and Saliency Map Interpretability"; arXiv:1905.04172v1 [stat.ML] 10 May 2019

[74] Been Kim et al. “Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)”. In: International conference on machine learning. PMLR, 2018, pp. 2668–2677.

[75] Zhang, R., Madumal, P., Miller, T., Ehinger, K.A., Rubinstein, B.I.: Invertible concept-based explanations for cnn models with non-negative concept activation vectors. In: Proc. AAAI Conf. Artificial Intelligence. pp. 11682–11690 (2021)

[76] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: Deep learning for interpretable image recognition. NeurIPS, 32, 2

[77] Kexin Pei et. al.; "DeepXplore: Automated Whitebox Testing of Deep Learning Systems"; arXiv:1705.06640v4 [cs.LG] 24 Sep 2017

[78] Youcheng Sun et. al.; "Testing Deep Neural Networks"; arXiv:1803.04792v4 [cs.LG] 15 Apr 2019

[79] Shogo Tokui et. al.; "NEURECOVER: Regression-Controlled Repair of Deep Neural Networks with Training History"; arXiv:2203.00191v2 [cs.LG] 4 Mar 2022

[80] James Kirkpatrick et. al.; "Overcoming catastrophic forgetting in neural networks"; https://www.pnas.org/doi/full/10.1073/pnas.1611835114

[81] Prakhar Kaushik et. al.; "Understanding Catastrophic Forgetting and Remembering in Continual Learning with Optimal Relevance Mapping"; arXiv:2102.11343v1 [cs.LG] 22 Feb 2021

[82] A. Tharwat, “Classification assessment methods.” Applied Computing and Informatics, Volume 17, Issue 1, 30, 2020.

[83] C. Sammut and G. I. Webb, Encyclopedia of machine learning: Springer Science & Business Media, 2011

[84] ISO/IEC TR 24027:2021, *Information technology — Artificial intelligence (AI) — Bias in AI systems and AI aided decision making*

[85] ISO/TR 4804:2020, *Road vehicles — Safety and cybersecurity for automated driving systems — Design, verification and validation*

[86] ISO 26262-8:2018, *Road vehicles — Functional safety — Part 8: Supporting processes*

© ISO 2024 – All rights reserved

[10] ISO 26262-6:2018, *Road vehicles — Functional safety — Part 6: Product development at the software level*

[87] Kuhn, D. Richard, Raghu N. Kacker, and Yu Lei. Introduction to combinatorial testing. CRC press, 2013.

[88] ISO/IEC 22989, *Information technology — Artificial intelligence — Artificial intelligence concepts and terminology*

[89] ISO 26262-2:2018, *Road vehicles — Functional safety — Part 2: Management of functional safety*

[90] ZHANG, Q. AND ZHU, S.-C., "Visual Interpretability for Deep Learning: a Survey", 2018, https://arxiv.org/abs/1802.00614

[91] ISO/TR 4804:2020, *Road vehicles — Safety and cybersecurity for automated driving systems — Design, verification and validation*

[92] ISO/IEC TR 24029-1:2021, *Artificial Intelligence (AI) — Assessment of the robustness of neural networks — Part 1: Overview*

[5] ISO 26262-10:2018, *Road vehicles — Functional safety — Part 10: Guidelines on ISO 26262*

[93] Sina Mohseni, Mandar Pitale, Vasu Singh, Zhangyang Wang: "Practical Solutions for Machine Learning Safety in Autonomous Vehicles", Safe AI 2020

[94] Practical Solutions for Machine Learning Safety in Autonomous Vehicles

[95] MOLNAR, C., "A Guide for Making Black Box Models Explainable, 2019, https://christophm.github.io/interpretable-ml-book/

[96] Shai Shalev-Shwartz and Amnon Shashua. On the Sample Complexity of End-to-end Training vs. Semantic Abstraction Training. arXiv:1604.06915, 2016, https://doi.org/10.48550/arXiv.1604.06915

[97] IEC 61508-4:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 4: Definitions and abbreviations (see Functional Safety and IEC 61508)*

[98] Koopmann, Tjark, Christian Neurohr, Lina Putze, Lukas Westhofen, Roman Gansch, and Ahmad Adee. 2022. “Grasping Causality for the Explanation of Criticality for Automated Driving.” arXiv preprint arXiv:2210.15375.

[99] arXiv:2302.10588. Salay, Rick, Matt Angus, and Krzysztof Czarnecki. 2019. “A safety analysis method for perceptual components in automated driving.” 2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE). 24–34.

[100] IEC 61508-4:2010, *Functional safety of electrical/electronic/programmable electronic safety-related systems - Part 4: Definitions and abbreviations (see Functional Safety and IEC 61508)*

[101] LAPUSCHKIN, S., WÄLDCHEN, S., BINDER, A., MONTAVON, G., SAMEK, W., and MÜLLER, K.-R., "Unmasking Clever Hans predictors and assessing what machines really learn", 2019, In: Nature Communications 1096 (2019), https://www.nature.com/articles/s41467-019-08987-4

[102] Dong, Yi, Wei Huang, Vibhav Bharti, Victoria Cox, Alec Banks, Sen Wang, Xingyu Zhao, Sven Schewe, and Xiaowei Huang. 2022. “Reliability Assessment and Safety Arguments for Machine Learning

© ISO 2024 – All rights reserved

Components in System Assurance." ACM Transactions on Embedded Computing Systems (ACM New York, NY).

[103] Acar Celik, Esra, Carmen Cârlan, Asim Abdulkhaleq, Fridolin Bauer, Martin Schels, and Henrik J. Putzer. 2022. "Application of STPA for the Elicitation of Safety Requirements for a Machine Learning-Based Perception Component in Automotive." Computer Safety, Reliability, and Security: 41st International Conference, SAFECOMP 2022, Munich, Germany, September 6–9, 2022, Proceedings. 319–332.

© ISO 2024 – All rights reserved