Safe Reinforcement Learning and the Future of ISO 26262

Jonas Riebel & Max Ulmke

Summer Semester 2018
Agenda

1. Introduction
2. How to combine Machine Learning and ISO 26262?
3. Solution Approaches
4. Conclusion and Outlook
Agenda

1. Introduction
   1. What is safety?
   2. Functional Safety
   3. ISO26262
   4. ASIL
   5. Changes from first to second edition

2. How to combine machine learning and ISO 26262?

3. How to move on from here?

4. Conclusion and Outlook
How to prove that it is safe?

Source: Source: https://www.thatcham.org/
Safety of technical systems

• What is safety?
  Safety: „Absence of unreasonable risk“ [ISO26262]

• Risk must be below a certain limit
  “Risk: combination of the probability of occurrence of harm and the severity of that harm” [ISO26262]
Safety of technical systems

• What is safety?
  Safety: „Absence of unreasonable risk“ [ISO26262]

• Risk must be below a certain limit
  “Risk: combination of the probability of occurrence of harm and the severity of that harm” [ISO26262]
Safety of technical systems

• What is safety?
  Safety: „Absence of unreasonable risk“ [ISO26262]

• Risk must be below a certain limit
  “Risk: combination of the probability of occurrence of harm and the severity of that harm” [ISO26262]

Source: Lecture Advanced Deep Learning for Robotics, Bäuml, SS18, TUM

Adversarial Example
Safety of technical systems

• What is safety?
  Safety: „Absence of unreasonable risk“ [ISO26262]

• Risk must be below a certain limit
  “Risk: combination of the probability of occurrence of harm and the severity of that harm” [ISO26262]

Source: https://inews.co.uk/essentials/lifestyle/cars/car-news/drivers-doubt-future-autonomous-cars/

Example for Misuse of a level 2 car.
Safety of technical systems

• What is safety?
  Safety: „Absence of unreasonable risk“ [ISO26262]

• Risk must be below a certain limit
  “Risk: combination of the probability of occurrence of harm and the severity of that harm” [ISO26262]

Failed Takata Airbag

Functional safety

• Potential failure of a system in different situation
• Goal:
  • show potential hazards and classify them
  • Reduction of hazards caused by E/E systems
• Outcome:
  • safety integrity level SIL/ASIL
  • Definition of processes and countermeasure to prevent failures
  • Regulated by standards
Functional safety (ISO 26262)

- Functional Safety: “absence of unreasonable risk due to hazards caused by malfunctioning behavior of E/E systems”

- ISO 26262: Road vehicles – functional safety
  - International standard
  - Scope: functional safety of E/E components in series production vehicles up to a mass of 3.5 tons
  - Specialization of the IEC 61508 for the automotive sector: Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related Systems

Relevant for all advanced driver assistance systems (ADAS)

[ISO26262, 2011]
ISO26262

1. Vocabulary

2. Management of functional safety

- 2-6 Overall safety management
- 2-8 Safety management during item development
- 2-7 Safety management after release for production

3. Concept phase

- 3-6 Item definition
- 3-6 Initiation of the safety lifecycle
- 3-7 Hazard analysis and risk assessment
- 3-8 Functional safety concept

4. Product development: system level

- 4-5 Initiation of product development at the system level
- 4-6 Specification of the technical safety requirements
- 4-7 System design
- 4-8 Item integration and testing

5. Product development: hardware level

- 5-5 Integration of product development at the hardware level
- 5-6 Specification of hardware safety requirements
- 5-7 Hardware design
- 5-8 Hardware architectural metrics

6. Product development: software level

- 6-5 Integration of product development at the software level
- 6-6 Specification of software safety requirements
- 6-7 Software architectural design
- 6-8 Software unit design and implementation
- 6-9 Software unit testing
- 6-10 Software integration and testing
- 6-11 Verification of software safety requirements

8. Supporting processes

- 8-5 Interfaces within distributed developments
- 8-6 Specification and management of safety requirements
- 8-7 Configuration management
- 8-8 Change management
- 8-9 Verification
- 8-10 Documentation
- 8-11 Qualification of software tools
- 8-12 Qualification of software components
- 8-13 Qualification of hardware components
- 8-14 Proven in use argument

9. ASIL-oriented and safety-oriented analyses

- 9-5 Requirements decomposition with respect to ASIL tailoring
- 9-6 Criteria for coexistence of elements
- 9-7 Analysis of dependent failures
- 9-8 Safety analyses

10. Guidelines on ISO 26262 (Informative)

Source: ISO26262
ISO26262

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2-5 Overall safety management</td>
<td>4. Product development at the system level</td>
</tr>
<tr>
<td>3-6 Initiation of the safety lifecycle</td>
<td>4-8 System design</td>
</tr>
<tr>
<td>3-7 Hazard analysis and risk assessment</td>
<td>4-9 Safety validation</td>
</tr>
<tr>
<td>3-8 Functional safety concept</td>
<td>4-10 Item integration and testing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Concept phase</th>
<th>4. Product development at the system level</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-9 Availability</td>
<td>4-11 Verification of software safety requirement</td>
</tr>
<tr>
<td>3-10 Reliability</td>
<td>4-12 Verification of hardware safety requirement</td>
</tr>
<tr>
<td>3-11 Maintainability</td>
<td>4-13 Verification of system safety requirement</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5. Product development at the hardware level</th>
<th>6. Product development at the software level</th>
</tr>
</thead>
<tbody>
<tr>
<td>5-1 Definition of the system</td>
<td>6-1 Design and implementation of the software</td>
</tr>
<tr>
<td>5-2 Requirements and design</td>
<td>6-2 Software validation and testing</td>
</tr>
<tr>
<td>5-3 Integration of the hardware</td>
<td>6-3 Software integration and testing</td>
</tr>
<tr>
<td>5-4 Construction and testing</td>
<td>6-4 System safety validation and testing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7. Production and operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>7-1 Production and operation of the system</td>
</tr>
<tr>
<td>7-2 Maintenance and repair of the system</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8. Supporting processes</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-1 Planning and management of the supporting processes</td>
</tr>
<tr>
<td>8-2 Change management</td>
</tr>
<tr>
<td>8-3 Verification</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9. ASIL-oriented and safety-oriented analyses</th>
</tr>
</thead>
<tbody>
<tr>
<td>9-1 ASIL definition</td>
</tr>
<tr>
<td>9-2 Safety oriented analysis</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>10. Guideline on ISO 26262</th>
</tr>
</thead>
<tbody>
<tr>
<td>10-1 Overview of the guideline</td>
</tr>
<tr>
<td>10-2 Requirements and definition</td>
</tr>
</tbody>
</table>

Source: Lecture Fahrerassistenzsysteme, Prof. Lienkamp, WS17, TUM
ISO26262

1. Vocabulary

2. Management of functional safety

3. Concept phase

4. Product development at the system level

7. Production and operation

5. Product development at the hardware level

6. Product development at the software level

8. Supporting processes

9. ASIL-oriented and safety-oriented analyses

10. Guideline on ISO 26262

Source: Lecture Fahrerassistenzsysteme, Prof. Lienkamp, WS17, TUM
ISO26262 - HARA

Hazard and Risk Analysis with ISO26262:

1. Item Definition
2. Situation analysis
3. Hazard identification
4. Classification of hazardous events
5. Determination of ASIL
6. Determination of safety goals

Source: Henriksson et al., 2018

[ISO26262, 2011]  [Henriksson et al., 2018]
ISO26262 - HARA

Hazard and Risk Analysis with ISO26262:

1. Item Definition
2. Situation analysis
3. Hazard identification
4. Classification of hazardous events
5. Determination of ASIL
6. Determination of safety goals

Source: Henriksson et al., 2018

[ISO26262, 2011] [Henriksson et al., 2018]
## ASIL Level

### Severity

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>S0 No injuries</td>
</tr>
<tr>
<td></td>
<td>S1 Light to moderate injuries</td>
</tr>
<tr>
<td></td>
<td>S2 Severe to life-threatening (survival probable) injuries</td>
</tr>
<tr>
<td>High</td>
<td>S3 Life threatening (survival uncertain) to fatal injuries</td>
</tr>
</tbody>
</table>

[ISO26262, 2011]
ASIL Level

Severity

Low
- S0
  - E1 – E4

S1
  - E1
  - E2
  - E3
  - E4

S2
  - E1
  - E2
  - E3
  - E4

High
- S3
  - E1
  - E2
  - E3
  - E4

Exposure

[ISO26262, 2011]
ASIL Level

Severity

Exposure

Controllability

High

Low

C1

C2

C3

[ISO26262, 2011]
ASIL Level

Severity       Exposure       Controllability

Low

S0

E1 – E4

QM
QM
QM

Low

S1

E1

QM
QM

E2

QM
QM

E3

QM
ASIL A

E4

ASIL B
QM

High

S2

E1

QM
QM

E2

QM
QM

E3

QM
ASIL A

E4

ASIL A
ASIL B

S3

E1

QM
QM

E2

QM
QM

E3

ASIL A
ASIL A

E4

ASIL B
ASIL C

[ISO26262, 2011]
ASIL Level

Severity

Low
S0
E1 – E4

S1
E1
E2
E3
E4

S2
E1
E2
E3
E4

High
S3
E1
E2
E3
E4

Exposure

Controllability

High
C1
C2
C3

Controllability

Low

Severity

Low
S0
E1 – E4

S1
E1
E2
E3
E4

S2
E1
E2
E3
E4

High
S3
E1
E2
E3
E4

Exposure

ASIL C

Severity

Low
S0
E1 – E4

S1
E1
E2
E3
E4

S2
E1
E2
E3
E4

High
S3
E1
E2
E3
E4

Exposure

[ISO26262, 2011]
# ASIL Level

## Severity

<table>
<thead>
<tr>
<th>S0</th>
<th>S1</th>
<th>S2</th>
<th>S3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>High</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>

## Exposure

<table>
<thead>
<tr>
<th>C1</th>
<th>C2</th>
<th>C3</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>Low</td>
<td>Low</td>
</tr>
</tbody>
</table>

## Controllability

<table>
<thead>
<tr>
<th>QM</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASIL A</td>
</tr>
<tr>
<td>ASIL B</td>
</tr>
<tr>
<td>ASIL C</td>
</tr>
<tr>
<td>ASIL D</td>
</tr>
</tbody>
</table>

### Germany: 732,9 Billion KM were driven in 2017

<table>
<thead>
<tr>
<th>ASIL Level</th>
<th>Failure in Time = $10^{-9}$ failures / hour</th>
<th>1 Failure per x km (~50km/h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASIL A</td>
<td>- Not defined in ISO</td>
<td></td>
</tr>
<tr>
<td>ASIL B</td>
<td>1000 FIT</td>
<td>50.000.000 km</td>
</tr>
<tr>
<td>ASIL C</td>
<td>100 FIT</td>
<td>500.000.000 km</td>
</tr>
<tr>
<td>ASIL D</td>
<td>10 FIT</td>
<td>5.000.000.000 km</td>
</tr>
</tbody>
</table>

[ISO26262, 2011]
# Changes to 2nd edition

## 1st Edition (2011)

<table>
<thead>
<tr>
<th>1. Vocabulary</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. Management of functional safety</td>
</tr>
<tr>
<td>3. Concept phase</td>
</tr>
<tr>
<td>4. Product development at the system level</td>
</tr>
<tr>
<td>5. Product development at the hardware level</td>
</tr>
<tr>
<td>6. Product development at the software level</td>
</tr>
<tr>
<td>7. Production and operation</td>
</tr>
<tr>
<td>8. Supporting processes</td>
</tr>
<tr>
<td>9. ASIL-oriented and safety-oriented analyses</td>
</tr>
<tr>
<td>10. Guideline on ISO 26262</td>
</tr>
</tbody>
</table>

Source: Lecture Fahrerassistenzsysteme, Prof. Lienkamp, WS17, TUM


- Removal of 3.5t weight limit
- New part 11 on semiconductors
- Guidelines on the application of ISO 26262 to semiconductors
- New qualification methods for ASIL
- Updates to the PMHF (Probabilistic Metric for Hardware Failure) equation and verification of safety analysis
- No changes regarding ML

[Schloeffel and Dailey, ]
Agenda

1. Introduction

2. How to combine Machine Learning and ISO 26262?
   1. What is safe in regards to AD?
   2. Possible Applications of ISO26262 to ML
   3. Conflicts between ISO26262 and ML

3. How to move on from here?

4. Conclusion and Outlook
What is safe in regards to AD?

• What is “safe enough” for AD?
  • No accidents at all (“Absence of unreasonable risk...”)
  • (significant) higher safety than today (lower accident rate)

• 5 billion test miles necessary for real-world verification
  • How do we define test cases (border cases)?
  • What is correct behaviour in tests?

[Bates, 2017] [Dailey, 2018]
Possible Applications of ISO26262 to ML

- 34 of 75 ISO SW-Development techniques at unit level
- black box techniques are “ok”
- “walk through” can be “adapted”
- Code oriented / white box techniques can not be used

Source: Lecture Fahrerassistenzsysteme, Prof. Lienkamp, WS17, TUM

[Salay et al., 2017]
What about incomplete training sets?

What about new types of hazards?

Which SW techniques to use?
Conflicts

- Identifying hazards
- Fault and failure modes
- The use of training sets
- The level of ML usage
- Required software techniques

[Salay et al., 2017]
Conflicts

• Identifying hazards
  • ML can create new types of hazards
  • Complex behavioral interactions between humans and ML
    • Overestimation of the ML performance
    • Human has not the optimal level of employment
    • Reduced human skill level and situation awareness
  • RL plays with the reward function or has unintended behavior through wrong reward function

• Fault and failure modes
• The use of training sets
• The level of ML usage
• Required software techniques

Yerkes-Dodson-Law (1908):
Source: Lecture Fahrerassistenzsysteme, Prof. Lienkamp, WS17, TUM

[Salay et al., 2017]
RL plays with the reward function

Source: https://blog.openai.com/faulty-reward-functions/
Conflicts

• Identifying hazards
• Fault and failure modes
  • Specific fault types and failure modes for ML
    • Network topology
    • Learning algorithm
    • Training set
• The use of training sets
• The level of ML usage
• Required software techniques

Source: https://leonardoaraujosantos.gitbooks.io/artificial-intelligence/content/convolutional_neural_networks.html

[Salay et al., 2017]
Conflicts

- Identifying hazards
- Fault and failure modes
- The use of training sets
  - Inherently incomplete data sets
  - Correct by construction with respect to the training set
  - Unspecifiable functionality (like perception)
- The level of ML usage
- Required software techniques

Imagenet Error Rates, Source: https://c1.staticflickr.com/5/4162/33621365014_fe35be452a_b.jpg

[Salay et al., 2017]
Conflicts

- Identifying hazards
- Fault and failure modes
- The use of training sets
- The level of ML usage
  - End-to-end learning is critical
  - Lack of transparency of ML components
- Required software techniques

Source: https://devblogs.nvidia.com/deep-learning-self-driving-cars/

[Salay et al., 2017]
Conflicts

• Identifying hazards
• Fault and failure modes
• The use of training sets
• The level of ML usage

• Required software techniques
  • Assumption that code is implemented using an imperative programming language
  • Difficult to use with ML, but also with Functional or logic programming, etc.

Source: Salay et al., 2017
Agenda

1. Introduction
2. How to combine Machine Learning and ISO 26262?
3. How to move on from here?
   1. Recommendations for applying ISO26262 to ML
   2. Usage of a new standard
      1. Pegasus
      2. SOTIF
   3. Radical new approach
4. Conclusion and Outlook
Recommendations for applying ISO26262 to ML

Changes to ISO26262

• Consider ML specific hazards

How to use ML

[Varshney, 2016] [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262

- Consider ML specific hazards
- ML Lifecycle techniques

How to use ML

[Varshney, 2016]   [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262

- Consider ML specific hazards
- ML Lifecycle techniques
- Partial specifiable functionality

How to use ML

[Varshney, 2016]  [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262

• Consider ML specific hazards
• ML Lifecycle techniques
• Partial specifiable functionality
• Fault tolerance strategies for software

How to use ML

[Varshney, 2016]  [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262

• Consider ML specific hazards
• ML Lifecycle techniques
• Partial specifiable functionality
• Fault tolerance strategies for software
• Intent based SW requirements

How to use ML

[Varshney, 2016]  [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262

• Consider ML specific hazards
• ML Lifecycle techniques
• Partial specifiable functionality
• Fault tolerance strategies for software
• Intent based SW requirements

How to use ML

• Modular ML usage

[Varshney, 2016] [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262
- Consider ML specific hazards
- ML Lifecycle techniques
- Partial specifiable functionality
- Fault tolerance strategies for software
- Intent based SW requirements

How to use ML
- Modular ML usage
- Human interpretable models

[Varshney, 2016] [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262

• Consider ML specific hazards
• ML Lifecycle techniques
• Partial specifiable functionality
• Fault tolerance strategies for software
• Intent based SW requirements

How to use ML

• Modular ML usage
• Human interpretable models
• Safety Reserves

[Varshney, 2016]  [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262

• Consider ML specific hazards
• ML Lifecycle techniques
• Partial specifiable functionality
• Fault tolerance strategies for software
• Intent based SW requirements

How to use ML

• Modular ML usage
• Human interpretable models
• Safety Reserves
• Reject option

[Varshney, 2016]  [Salay et al., 2017]
Recommendations for applying ISO26262 to ML

Changes to ISO26262
- Consider ML specific hazards
- ML Lifecycle techniques
- Partial specifiable functionality
- Fault tolerance strategies for software
- Intent based SW requirements

How to use ML
- Modular ML usage
- Human interpretable models
- Safety Reserves
- Reject option
- Open Source / Data

https://www.123rf.com/stock-photo/hands_reaching_out.html
[Varshney, 2016] [Salay et al., 2017]
Usage of an new standard

+ No constraints of old standard with old software paradigms
+ Everything could be rethought and fitted to ML
+ E.g. open source norm (maintained by a group of e.g. Automotive developers)

- Using a new standard correlates with retraining the companies employees
- Difficult to make it a standard
Pegasus

PEGASUS RESEARCH PROJECT
SECURING AUTOMATED DRIVING EFFECTIVELY.

PEGASUS delivers the standards for the automation of the future. With the PEGASUS joint project, promoted by the Federal Ministry for Economic Affairs and Energy (BMWi), key gaps in the field of testing of highly-automated driving functions will be concluded by the middle of 2019.

Source: https://www.pegasusprojekt.de/en/

Fundet by the german governement (BMWi) for developing automated driving standards
**SOTIF (Safety Of The Intended Functionality) – not yet released**

- Will provide guidelines for Level-0, Level-1, and Level-2 autonomous drive (AD) vehicles.
- Focus on the functional safety of ADAS-related hazards caused by “normal operation” of the sensors.
- Questions to be answered by SOTIF:
  - Details on advanced concepts of the AD architecture
  - How to evaluate SOTIF hazards that are different from ISO 26262 hazards?
  - How to identify and evaluate scenarios and trigger events?
  - How to reduce SOTIF related risks?
  - How to verify and validate SOTIF related risks?
  - The criteria to meet before releasing an autonomous vehicle.
Radical new approach

"Wenn ein Problem von den Testfahrern festgestellt wurde, wird das anschließend behoben und das Ganze geht von vorne los. Aber wenn Sie etwas behoben haben, müssen Sie sehen, dass an einer anderen Stelle die Funktion genau funktioniert wie vorher. Das ist schon ein bisschen tricky, aber das ist eben die Pionierleistung", sagt Bereczki (Regulierungsexperte bei Audi).

Translated:

‘If a problem has been identified by the test driver, it will be fixed and the whole thing will start again. But if you have fixed something, you have to see that in another place the function works exactly as before. That's a bit tricky, but that's just the pioneering work’ says Bereczki. (Regulationexpert at Audi).

Every manufacturer must solve this task for themselves, since there are not yet defined test scenarios for this.
Radical new approach

• Approach similar to Aerospace Industry
• Every failure has to be reported.
• Faults have to be investigated
• Resulting new test cases will be added
• Continuous improved test case database

→ Continuous Improvement and cooperation of Automotive industry


[Bates, 2017]
Agenda

1. Introduction
2. How to combine Machine Learning and ISO 26262?
3. How to move on from here?
4. Conclusion and Outlook
Conclusion

• ISO26262
  • ISO26262 and functional safety fit partially and with constraints to ML techniques
  • Changes on the standard are required (e.g. separation between fully and partially specified tasks)
  • The allowed usage of ML techniques will be restricted

• Other Standards
  • SOTIF: guidance for assuring that an autonomous vehicle functions and acts safely during normal operation
  • Pegasus: Close key gaps in the field of testing of highly-automated driving functions will be concluded by the middle of 2019.

• In general, ML is only applicable under constraints in safety critical (no end-to-end, safe guards,..)
  → Applicability of Reinforcement Learning even worse (more parameter like the correct reward function have to be estimated)
Outlook - Quotes from industry

- **December 2016:** “We’re going to end up with complete autonomy, and I think we will have **complete autonomy in approximately two years.**” Elon Musk
  
  https://electrek.co/2015/12/21/tesla-ceo-elon-musk-drops-prediction-full-autonomous-driving-from-3-years-to-2/

- **March 2017:** “**We are on the way to deliver a car in 2021 with level 3, 4 and 5**” BMW senior vice president for Autonomous Driving – Frickenstein
  

- **March 2018:** “**Level 3 will be achieved by 2021, but Level 4 is one of the biggest challenges facing the auto industry**”, Elmar Frickenstein. (translated)
  
Outlook – own thoughts

- New simulation environments and massive data sets (ongoing)
- End of Pegasus Project (~2020)
- New standards like SOTIF (~2020)
- ISO 26262 3rd edition for level 3 car (~2022)
Resources


Thank you for your attention

megapope

self driving cars aren’t even hard to make lol just program it not to hit stuff

ronpaulhdwallpapers

if(goingToHitStuff) {
    dont();
}

Source: megapope