# ASSURANCE FOR MACHINE LEARNING SYSTEMS

# Abstract

Abstract here (no more than 300 words)

# Contents

# List of Figures

# List of Tables

# Notation, Definitions, and Abbreviations

## Abbreviations

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **ML** | Machine Learning |
| **AV** | Autonomous Vehicle |
| **ASIL** | Automotive Safety Integrity Level |
| **MDE** | Model Driven Engineering |
| **PDF** | Probability Distribution Function |
| **RL** | Reinforcement Learning |
| **GSN** | Goal Structuring Notation |

# Chapter 1

# Introduction

With new developments in Artificial Intelligence (AI) and ML, a growing number of research projects in this field and many companies have started utilizing these methods. ML methods are also used in many safety critical applications such as Autonomous Vehicles (AVs) and healthcare applications. Therefore, it is very important to have a clear perspective of the safety of such methods in these applications.

In some applications, an erroneous outcome of the ML model has a harmful impact on many lives, for example in medical diagnosis [14], loan approval [19], autonomous vehicles [18], and prison sentencing [6]. Despite the numerous research papers in this subject, there is still a need to delve deeper and understand the behavior of ML systems in safety critical applications.

One major drawback in using ML algorithms is that they are often treated as a black box and hence, using safety procedures for these methods is sometimes inapplicable [23]. In a review of automotive software safety methods [22], an analysis of ISO-26262 part-6 methods was performed with respect to safety of ML models. This assessment shows that about 40% of software safety methods do not apply to ML

models [22].

Safety specifications often assume that behaviour of a component is fully specified. Since the training sets used in ML methods are not necessarily complete, they violate this assumption, and some parts of the specification becomes not applicable to the ML components [22]. Most widely used ML frameworks such as Tensorflow [2] and Theano [3] employ a model driven approach in problem solving. Although model driven engineering approach has been successful in safety critical applications such as Automotive industry, the ML models cannot be guaranteed to operate in a safe manner.

There are two approaches with respect to ML and safety, first is to study safety of ML methods, algorithms, and processes and the second is to use ML methods to improve pre-existing safety assurance procedures. We will initially follow the first approach and review the literature for the methods applied to standardize and measure the safety of ML methods.

There are inherent performance metrics related to ML methods, such as accuracy and robustness, which can affect their applicability in safety critical applications. ML models can also be dependent to the domain they are trained [15]. In addition, other perturbations such as noise, natural and imaging artifacts can cause ML models to function less accurately [16].

Assurance cases have been successfully used in various industries to describe why a system can be trustfully used for a specific application [5].

A recent definition of safety assurance case is described in [8] as

> "A structured argument, supported by a body of evidence, that provides a compelling, comprehensible and valid case that a system is safe for a

given application in a given environment"

A structured argument is a [21]

"connected series of statements or reasons intended to establish a position...; a process of reasoning."

Reasons used in a structured argument can be considered as premises in logical terms and a conclusion can be drawn based on them. [21].

To obtain assurance for ML systems it is essential to understand the ML lifecycle. This lifecycle follows a spiral process model [9] and is comprised of four stages [5]

- Data Management(DM)
- Model Learning
- Model Verification(MV)
- Model Deployment

# Chapter 2

# Machine Learning

Machine learning algorithms can extract patterns and learn from data [11]. A brief definition of learning can be given as [20]

> "A computer program is said to learn from experience $E$ with respect to some class of tasks $T$ and performance measure $P$, if its performance at tasks in $T$, as measured by $P$, improves with experience $E$."

A task is the main objective of using an ML algorithm. For example, in an autonomous vehicle, driving the car is the task. A task is not the process of learning. Learning is used as a means to achieve an ability to accomplish a task [11]. With developments in ML methods, they have been applied to different tasks, some examples of tasks are classification, regression, transcription, machine translation, denoising [11].

The performance measure is used to quantify how successfully a task is accomplished, equivalently, number of erroneous outputs could be used as a way of indicating a method's performance.

Based on the above-stated definition, the ML algorithm undergoes an experience in the process of learning. This experience is generally classified into **unsupervised**, **supervised** and **reinforcement** learning.

Unsupervised learning finds the properties of the overall structure of the dataset. Clustering as an example of unsupervised learning, finds clusters within a dataset and assigns each data-point to one of them.

In supervised learning, on the other hand, data-points that the learning algorithm experiences have a label. This label acts as a guide for the ML algorithm. The term supervised arises from the fact that the labels instruct the algorithm what to do. Labels are unavailable in unsupervised learning and the ML system is responsible to make sense of the data independently [11].

Reinforcement learning (RL) algorithms experience an environment instead of a fixed dataset. The algorithm should learn how to maximize a reward function by taking an appropriate action [24]. The learner discovers this appropriate action by trying different actions and observing the value of the reward function. Actions not only affect the immediate reward, but can also change next actions' rewards. Trial and error search and delayed reward are two main characteristics of RL.

The learner, also known as the agent in RL terms, should have the capability to sense the state of the environment, take actions that can alter the state and also have a goal to reach by taking actions. These three aspects are included in the reward function used by the agent [24] more about Deep RL?

Evidently, ML algorithms need data to learn and function. A dataset can be described as a **design matrix**. Every row in the matrix contains an example, also known as data-point, and each column is a feature. Iris dataset is one of the first

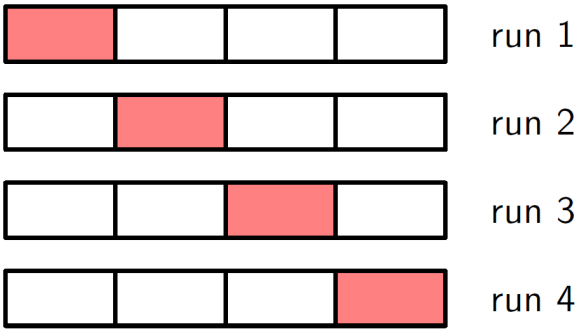


Figure 2.1: Cross validation for S=4 [7].

ones used in statistics and ML [12]. This dataset is comprised of 150 examples which have 4 features each. One example corresponds to one individual plant. Sepal length, sepal width, petal length and petal width are recorded as features of each plant [12]. This means that if $X$ is the matrix, we can say $\mathbf{X} \in R^{150\times 4}$

The ML model will ultimately be deployed and used in a real world situation, hence, we are interested in how well an ML model performs on the data it has not seen before, this is also known as **generalization**. A portion of dataset is therefore not used in the training process and reserved as a **test set**. The data used in the training process is accordingly referred to as the **training set** [11].

In some cases, the training and test datasets might be limited in size and to have a better generalization, it is necessary to use as much of the data for training as possible. In other words, there will be less data available to estimate the performance of the model. One solution for this situation is **cross-validation**. The entire dataset is split into $S$ subsets. In each run, $S-1$ subsets are used for training and one remaining subset is the test set. For the next run, a different test set is selected [7]. Figure 2.1 shows selection of subset for $S = 4$.

add about neural networks

# Chapter 3

# Safety and assurance

Safety is often defined as [13]:

> Absence of unreasonable risk.

An unreasonable risk is a [13]:

> Risk judged to be unacceptable in a certain context according to valid societal moral concepts.

Various safety standards have been developed for different industries and activities. Some examples are ISO 26262 for functional safety of road vehicles, DO-178C for aerospace industry, ISO 8124 for safety of toys, ISO 7164 for healthcare organization management.

## 3.1 Safety Assurance Case

A safety assurance case (Assurance case in short) justifies safety of a system by bringing a valid argument that a set of claims are justified, given that a set of assumptions

are fulfilled [10]. The purpose of using an assurance case is to communicate a clear, comprehensive, defensible argument that a system is safe to be used in a particular context [17] Assurance cases are comprised of five basic components: claims, arguments, evidence, justifications and assumptions. The most common use of assurance cases is to give assurance about system's functionality and properties to the parties which were not involved in the process of developing the system [1].

Assurance cases reason in a subjective manner, as compared to the logical proofs which consider an absolute truth. In other words, assurance cases are useful because the full range of a system's properties are not always representable in a logical formalization. Also, assurance cases may sometimes be disproved because the underlying logical theory used in them is not relevant [1].

Since assurance cases are considered artefacts, they inherit quality related properties of them such as: the structure of its content, semantic features such as completeness, creation and maintenance. The conclusions of the assurance case should also be stated clearly with clear level of uncertainty [1]. more from 15020 about assurance cases?

## 3.2 Goal Structuring Notation

When the safety assurance case is more complex in nature, textual representation suffers to express the case in a clear and understandable way. Figure 3.1 shows an example of such problem where the English structure of the argument is hard to understand. Having multiple cross references is specially difficult to capture in text [17].

```
  For hazards associated with warnings,
the assumptions of [7] Section 3.4
associated with the requirement to
present a warning when no equipment
failure has occurred are carried
forward. In particular, with respect to
hazard 17 in section 5.7 [4] that for
test operation, operating limits will
need to be introduced to protect against
the hazard, whilst further data is
gathered to determine the extent of the
problem.
```

Figure 3.1: Problems associated with textual representation [17].

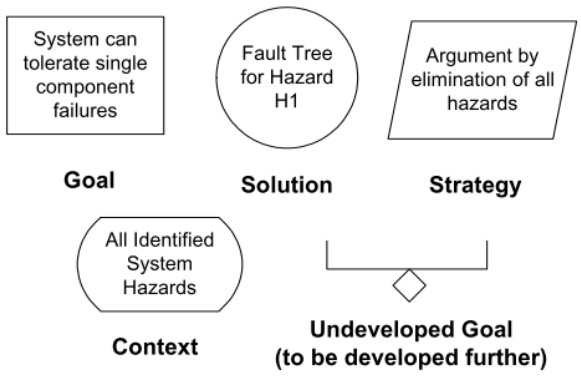


Figure 3.2: Basic elements of a GSN [17].

The Goal Structuring Notation(GSN) is a graphical notation for safety argumentation. A GSN explicitly represents elements of a safety argument and the relationships among these components. For example, how requirements are supported by claims or how claims are supported by evidence or how the case has a defined context [17]. Figure 3.2 depicts basic building blocks of a GSN with example instances of each element.

When these blocks are connected they make a "goal structure". The goal structure is then used show how goals (claims about the system) can be split into sub-goals successively until the sub-goal can be directly supported by available evidence.

write about CAE(Claims Arguments Evidence)

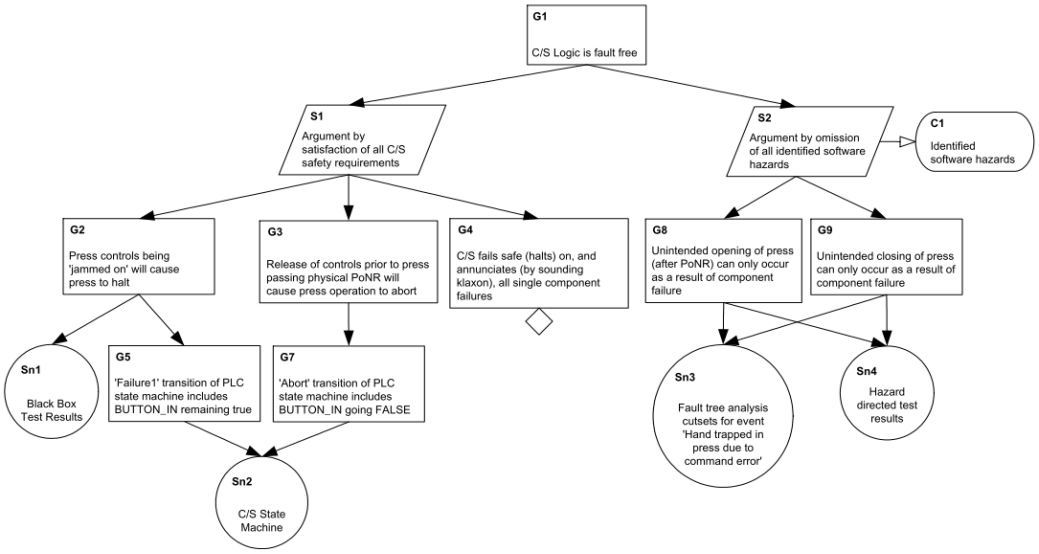


Figure 3.3: An example of a goal structure [17].

# Chapter 4

# Literature Review

In [4], five major research problems associated with unsafe behavior of ML models is presented. They can be summarized as

1. Avoiding Negative Side Effects: How to ensure that the model will not disturb the environment while pursuing its goals, e.g. can a cleaning robot knock over a vase because it can clean faster by doing so? Can we do this without manually specifying everything the robot should not disturb [4]?

2. Avoiding Reward Hacking: How to ensure that the model does not avoid situations to achieve a higher reward. For example, if we reward the robot for achieving an environment free of messes, it might disable its vision so that it won't find any messes, or cover over messes with materials it can't see through, or simply hide when humans are around so they can't tell it about new types of messes [4].

3. Scalable Oversight: How to ensure the model respects the parts of the objective function that are expensive to evaluate and makes a safe approximation of these

parts. For example, in the cleaning robot example, if the user is happy with the cleaning quality is an expensive objective function, but it can be approximated to presence of any dirt on the floor when the user arrives [4].

4. Safe Exploration: How to ensure that the ML model explorations are safe. For example, the robot should experiment with mopping strategies, but putting a wet mop in an electrical outlet is a very bad idea [4].

5. Robustness to Distributional Shift: How to ensure that the model performs robustly if the environment shifts from the training environment. For example, strategies a cleaning robot learns for cleaning an office might be dangerous on a factory work-floor [4].

# Chapter 5

# Conclusion

Every thesis also needs a concluding chapter

# Bibliography

[1] 15026-1-2019 - ISO/IEC/IEEE International Standard - Systems and software engineering–Systems and software assurance –Part 1:Concepts and vocabulary. *ISO/IEC/IEEE 15026-1:2019(E)*, pages 1–38, 2019.

[2] Martín Abadi, Michael Isard, and Derek G Murray Google Brain. A Computational Model for TensorFlow An Introduction.

[3] Rami Al-Rfou, Guillaume Alain, Amjad Almahairi, Christof Angermueller, Dzmitry Bahdanau, Nicolas Ballas, Frédéric Bastien, Justin Bayer, Anatoly Belikov, Alexander Belopolsky, Yoshua Bengio, Arnaud Bergeron, James Bergstra, Valentin Bisson, Josh Bleecher Snyder, Nicolas Bouchard, Nicolas Boulanger-Lewandowski, Xavier Bouthillier, Alexandre De Brébisson, Olivier Breuleux, Pierre-Luc Carrier, Kyunghyun Cho, Jan Chorowski, Paul Christiano, Tim Cooijmans, Marc-Alexandre Côté, Myriam Côté, Aaron Courville, Yann N Dauphin, Olivier Delalleau, Julien Demouth, Guillaume Desjardins, Sander Dieleman, Laurent Dinh, Mélanie Ducoffe, Vincent Dumoulin, Samira Ebrahimi Kahou, Dumitru Erhan, Ziye Fan, Orhan Firat, Mathieu Germain, Xavier Glorot, Ian Goodfellow, Matt Graham, Caglar Gulcehre, Philippe Hamel, Iban Harlouchet, Jean-Philippe Heng, Balázs Hidasi, Sina Honari, Arjun Jain, Sébastien Jean, Kai

Jia, Mikhail Korobov, Vivek Kulkarni, Alex Lamb, Pascal Lamblin, Eric Larsen, César Laurent, Sean Lee, Simon Lefrancois, Simon Lemieux, Nicholas Léonard, Zhouhan Lin, Jesse A Livezey, Cory Lorenz, Jeremiah Lowin, Qianli Ma, Pierre-Antoine Manzagol, Olivier Mastropietro, Robert T Mcgibbon, Roland Memisevic, Bart Van Merriënboer, Vincent Michalski, Mehdi Mirza, Alberto Orlandi, Christopher Pal, Razvan Pascanu, Mohammad Pezeshki, Colin Raffel, Daniel Renshaw, Matthew Rocklin, Adriana Romero, Markus Roth, Peter Sadowski, John Salvatier, François Savard, Jan Schlüter, John Schulman, Gabriel Schwartz, Iulian Vlad Serban, Dmitriy Serdyuk, Samira Shabanian, Etienne Simon, Sigurd Spieckermann, S Ramana Subramanyam, Jakub Sygnowski, Jérémie Tanguay, Gijs Van Tulder, Joseph Turian, Sebastian Urban, Pascal Vincent, Francesco Visin, Harm De Vries, David Warde-Farley, Dustin J Webb, Matthew Willson, Kelvin Xu, Lijun Xue, Li Yao, Saizheng Zhang, and Ying Zhang. Theano: A Python framework for fast computation of mathematical expressions (The Theano Development Team) *. Technical report.

[4] Dario Amodei, Chris Olah, Google Brain, Jacob Steinhardt, Paul Christiano, John Schulman, Openai Dan, and Mané Google Brain. Concrete Problems in AI Safety. Technical report.

[5] Rob Ashmore, Radu Calinescu, and Colin Paterson. Assuring the Machine Learning Lifecycle. *ACM Comput. Surv.*, 54(5):1–39, may 2021.

[6] Richard Berk and Jordan Hyatt. Machine Learning Forecasts of Risk to Inform Sentencing Decisions. *Source Fed. Sentencing Report.*, 27(4):222–228, 2015.

[7] C M Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[8] Robin Bloomfield and Peter Bishop. Safety and assurance cases: Past, present and possible future - An adelard perspective. In *Mak. Syst. Safer - Proc. 18th Safety-Critical Syst. Symp. SSS 2010*, pages 51–67. Springer London, 2010.

[9] Barry Boehm and Wilfred J Hansen. Spiral Development: Experience, Principles, and Refinements Spiral Development Workshop February 9, 2000. Technical report, 2000.

[10] Simon Burton, Lydia Gauerhof, and Christian Heinzemann. Making the Case for Safety of Machine Learning in Highly Automated Driving.

[11] Ian Goodfellow Courville, Yoshua Bengio, and Aaron. *Deep learning*, volume 29. MIT Press, 2016.

[12] R. A. FISHER. THE USE OF MULTIPLE MEASUREMENTS IN TAXONOMIC PROBLEMS. *Ann. Eugen.*, 7(2):179–188, sep 1936.

[13] International Organization for Standardization. *ISO 26262: Road Vehicles : Functional Safety*. ISO, 2018.

[14] Kenneth R Foster, Robert Koprowski, and Joseph D Skufca. Machine learning, medical diagnosis, and biomedical engineering research-commentary. Technical report, 2014.

[15] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *32nd Int. Conf. Mach. Learn. ICML 2015*, 2:1180–1189, sep 2015.

[16] Dan Hendrycks and Thomas Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. *arXiv*, mar 2019.

[17] Tim Kelly and Rob Weaver. The Goal Structuring Notation-A Safety Argument Notation.

[18] Philip Koopman and Michael Wagner. Challenges in Autonomous Vehicle Testing and Validation. Technical report, 2016.

[19] Stefan Lessmann, Bart Baesens, Hsin-Vonn Seow, and Lyn C Thomas. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *Eur. J. Oper. Res.*, 247:124–136, 2015.

[20] T M Mitchell. *Machine Learning*. McGraw-Hill international editions - computer science series. McGraw-Hill Education, 1997.

[21] Omg. An OMG® Structured Assurance Case Metamodel TM Publication Structured Assurance Case Metamodel (SACM) Version 2.1 OMG Document Number Release Date. Technical report, 2010.

[22] Rick Salay, Rodrigo Queiroz, and Krzysztof Czarnecki. An Analysis of ISO 26262: Using Machine Learning Safely in Automotive Software, sep 2017.

[23] Gesina Schwalbe and Martin Schels. A Survey on Methods for the Safety Assurance of Machine Learning Based Systems A Survey on Methods for the Safety Assurance of Machine Learning Based Systems. 10th European Congress on Embedded Real Time Software and Systems (ERTS A Survey on Methods for . Technical report, 2020.

[24] Richard S Sutton. Introduction: The challenge of reinforcement learning. In *Reinf. Learn.*, pages 1–3. Springer, 1992.