Assurance for Machine Learning systems

ASSURANCE FOR MACHINE LEARNING SYSTEMS

BY

MILAD HASSANI, B.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTING AND SOFTWARE

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

Master of Applied Science

Master of Applied Science (2021) McMaster University

(computing and software) Hamilton, Ontario, Canada

TITLE:	Assurance for Machine Learning systems
AUTHOR:	Milad Hassani B.Sc.
SUPERVISOR:	Dr. Richard Paige
NUMBER OF PAGES:	1, 9

Lay Abstract

A lay abstract of not more 150 words must be included explaining the key goals and contributions of the thesis in lay terms that is accessible to the general public.

Abstract

Abstract here (no more than 300 words)

Contents

List of Figures

List of Tables

Notation, Definitions, and Abbreviations

Notation

$A \leq B$	A is less than or equal to B

Definitions

Challenge

Abbreviations

AI	Artificial Intelligence
ML	Machine Learning
AV	Autonomous Vehicle
ASIL	Automotive Safety Integrity Level
MDE	Model Driven Engineering

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 [7], loan approval [11], autonomous vehicles [10], and prison sentencing [5]. 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 [14]. In a review of automotive software safety methods [13], 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 [13].

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 [13]. Most widely used ML frameworks such as Tensorflow [1] and Theano [2] 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 [8]. In addition, other perturbations such as noise, natural and imaging artifacts can cause ML models to function less accurately [9]. In [3], five major research problems associated with unsafe behaviour 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 [3]?

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 [3].

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 [3].

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 [3].

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 workfloor [3].

Assurance cases should be generated to describe why a system can be trustfully used for a specific application [4].

A recent definition of safety assurance case is described in [6] 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 [12]

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

These reasons can be considered as premises in logical terms and a conclusion can be drawn based on them[12].

this might need more expansion as to what are some of the examples of these premises and the assurance cases.

Chapter 2

Conclusion

Every thesis also needs a concluding chapter

Bibliography

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

[2] 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.

[3] 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.

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

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

[6] Robin Bloomfield and Peter Bishop. Safety and assurance cases: Past, present

and possible future - An adelard perspective. In *Making Systems Safer - Proceedings of the 18th Safety-Critical Systems Symposium, SSS 2010*, pages 51–67. Springer London, 2010.

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

[8] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *32nd International Conference on Machine Learning, ICML 2015*, 2:1180–1189, sep 2015.

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

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

[11] 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. *European Journal of Operational Research*, 247:124–136, 2015.

[12] 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.

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

[14] 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.