Proceedings Paper:
Johnson, Nikita Laura and Fang, Xinwei (2019) Three Reasons Why: Framing the Challenges of Assuring AI. In: Three Reasons Why: Framing the Challenges of Assuring AI.

Three Reasons Why: Framing the Challenges of Assuring AI

Xinwei Fang and Nikita Johnson

Department of Computer Science,
University of York, York, UK
{xinwei.fang, nikita.johnson}@york.ac.uk

Abstract. Assuring the safety of systems that use Artificial Intelligence (AI), specifically Machine Learning (ML) components, is difficult because of the unique challenges that AI presents for current assurance practice. However, what is also missing is an overall understanding of this multi-disciplinary problem space. In this paper, a model is given that frames the challenges into three categories which are aligned to the reasons why they occur. Armed with a common picture of where existing issues and solutions "fit-in", the aim is to help bridge cross-domain conceptual gaps and provide a clearer understanding to safety practitioners, ML experts, regulators and anyone involved in the assurance of a system with AI.

Keywords: Machine Learning · Safety · Assurance · Sensors.

1 Introduction

There are several advantages of adopting systems which use machine learning (ML) components. These advantages range from improving mobility in the automotive industry to innovative uses in industrial control and hazardous zones. It seems that very few safety-critical domains remain unaffected by this trend.

Whilst the scale and rate of adoption of ML is novel, the underlying ML technologies, such as Artificial Neural Networks, are not. Nor are many of the safety assurance challenges we face when using artificial intelligence - Rushby's 1988 report expertly explores the issues arising from attempting to assure knowledge-based AI, and presents some approaches to address these [10]. Thirty years later, many of the issues remain largely unresolved; one example is our poor understanding of software metrics that should take into account, not only the software behaviour in isolation, but consider the human developer (knowledge, skills and experience), the development and operational environment, and the objective of the product [10, p.26]. Without this understanding or theory of software development, our existing metrics applied to ML algorithms are still insufficient for safety assurance.

There have been several research advances and approaches developed to improve ML safety, such as - modelling safety assurance arguments for ML [3], characterising viewpoints for ML assurance [4], developing appropriate testing models and argument structures [7], creating models of how and when AI might become unsafe [11]. However, there remains no consensus on how to resolve the assurance challenge, namely - how do we *know* that a system with ML is safe. In addition, the introduction of ML components to a system not only creates new challenges, but acts as a force multiplier for the existing problems in safety assurance (such as the inheritance of context to subsystems, evidence sufficiency for claims, epistemic uncertainty introduced during design, *etc.*).

In order to assure the safety of ML we must first understand assurance and begin to build a picture of how these new and existing challenges and solutions relate to each other. To this end, this paper will briefly discuss views of assurance and current practice in **Section 2**; explore some of the specific differences between traditional systems and systems with ML components in **Section 3**; **Section 4** explores the three core challenges that are the subject of this paper; Followed by a dicussion in **Section 5**; lastly, **Section 6** will conclude the discussion by presenting a potential way forward for assurance of ML.

2 Assurance: Current Practice

At its core, assurance is concerned with managing uncertain negative outcomes. This is reflected in the definition of assurance in safety standards across several domains. Even though there is variance in the definitions, all of the standards approach assurance from at least one of three perspectives.

2.1 Assurance as an Outcome

This is the *reasoning* why a system is safe. It can exist in the minds of those developing the system; often as an argument and mental model of how safety works for that system. It is usually a requirement that this reasoning or *justified true belief in the safety of the system* be recorded for internal audit and external evaluation by regulatory bodies. It is represented and communicated through a combination of system artefacts, risk analysis models, test reports, justification reports and safety cases, *etc.*.

2.2 Assurance as a Process

This describes the steps required to *develop* and *record* the safety reasoning for a system. Whilst the *develop* part of assurance is concerned with risk reduction activities and good engineering (sometimes called *ensurance*), the *record* part of the process is concerned with systematically documenting the activities, and argumentation for building convincing reasoning. The result of this Assurance Process is the Outcome.

2.3 Assurance as a Relationship

This is the *relationship* that exists between the person making the assurance argument and the person whom they wish to persuade. There is an intuitive understanding of this when phrases such as *"I assure you that ..."* are used in everyday speech; however the relationship is not as obvious from current assurance practice and the standards. This is because there are implicit assumptions and shared understanding, *e.g.* when utilising a standard, of who is making an argument, and to whom it is being made.

Each of these perspectives present unique challenges when ML is incorporated into a system.

3 Differences Between Traditional and New

Having explored the existing assurance approaches in the previous section, the difference between traditional (non-autonomous system) and new systems (autonomous system) is discussed in this section.

Table 1. Differences Between AGV and SGV

AGV (traditional system)	**SGV** (new AI system)
Navigation along preplanned paths	Unplanned paths
Static map of environment	Dynamic environment modelling
Separation from humans and hazard zones	Interaction with humans
Linear *sense* function	Complex non-linear *sense* function
e.g. detect magnetic strip	*e.g.* detect "a person"
Programmed decision-making	Autonomous decision-making

Table 1 shows the differences between an Automated Guided Vehicle (AGV) and Self-Guided Vehicle (SGV). Pre-programmed AGVs have been manufactured and use for over thirty years, especially for applications where the tasks are simple and repetitive, such as moving stock in a warehouse. SGV are new systems because they allow for greater flexibility and capabilities through autonomous interaction with the environment. One of the major differences is in their *sensing* process. Both processes might have the sensing requirement not to collide with an object. However, for the added capability of interacting with humans, the SGV also needs to identify what a 'human' is.

To formulate the sensing process, let us denote S as the sensing state in real physical environment and X as the domain of sensors, and consider a sensing process as a function $\hat{S}()$: $X \rightarrow S$. Then, a data mapping from the sensor X can be formalised as $\hat{S}(X)$. In the AGV application, the sensed data $\hat{S}(X)$ is directly related to the sensing state S (the objects). For example, the output of a proximity sensor is directly related to the presence of objects. Therefore, the relationship between a sensing state and data can be represented as $\hat{S}(X) + \epsilon_s = S$, where the ϵ stands for uncertainties. We often say that data $\hat{S}(X)$ is ϵ_s accurate with respect to the sensing state S, denoted as $Q(\hat{S}(X), \epsilon_s)$. By minimising ϵ_s and analysing the cause of it, the uncertainty ϵ_s can be bounded. In that case, the requirements on data $\hat{S}(X)$ can be directly decomposed into requirements on sensor X, and evidence can be collected by testing sensor X.

However, in the SGV application, since no sensors can physically and directly sense a 'human', the sensed data $\hat{S}(X)$ is no longer directly related to the sensing state S (human). For example, a camera produces images which are RGB values. In order to identify a person from the images, a process of images is often needed. We symbolise such process as $F()$. As a result, the previous relationship can be written as $F(Q(\hat{S}(X), \epsilon_s)) + \epsilon_F = S$, where ϵ_s and ϵ_F are the accuracy for the sensing and the processing respectively. Since it is difficult for system developers to determine the $F()$, ML is the alternative that provides an approximation for $F()$. However, the $F()$ determined by ML is sensitive to many variables such as data distribution, sensor accuracy, or model parameters [2], which prevent the safety requirements propagating through them. This causes issues for safety assurance.

4 Understanding Challenges for Assuring ML

The objective of safety assurance is to bound uncertainty and build belief in the safety through various means, such as past experience, best practice and standards. Safety assurance often involves a number of tests such as code verification, timing, independence and formal tests to evaluate whether lower level requirements can be met. This is based on the assumption that the lower level safety requirements maintain the intent of higher level requirements through decomposition [6], however since requirements are not *decomposed* in the traditional sense with ML components, the assumptions that form the foundation for standards are violated. Current practice is not directly applicable [2]. In this section, three challenges related to assuring ML are identified. **Figure 1** illustrates these.

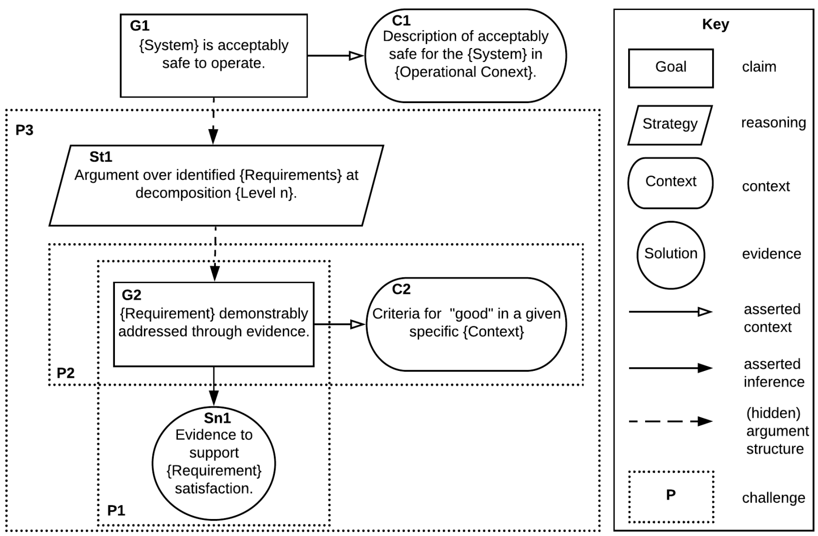

Fig. 1. Argument structure showing ML assurance challenges

4.1 Challenge 1 - Specifying tests without considering contexts (P1)

The existing safety standards require a system to undertake a number of tests (e.g. timing analysis). By passing those tests, evidence to support lower level requirements is provided. In traditional system satisfying lower level requirements leads to higher level requirements being satisfied because of the strong traceable decomposition and context inheritance. However, current standards were not designed for systems with ML, therefore it is possible for them to pass the tests, but behaviour to be unsafe[1]. For example, the issues of reward hacking in ML component is unrelated to how the software is coded [1].

[1] Note that this is true for traditional systems, however there is exponentially more uncertainty for ML system behaviour.

4.2 Challenge 2 - Specifying contexts without providing tests (P2)

The behaviour of a ML component is difficult to bound as it is sensitive to many variables as discussed in Section 3. There are many works available that try to bound the behaviours of a ML component [3,4,9]. Despite being from different perspectives (e.g. argue from performance level [9], functional viewpoints [4], and insufficiency [3]), they all provide a clear context that the behaviour of their ML components can be bounded. However, the challenge then becomes how to provide evidence to support their argument as no test options were given. For example, one of a lower level goal in [3] is '*The function is robust against distributional shift in the environment*'. Since it is not clear how evidence can be collected to support this requirement, the problem is still unsolved.

4.3 Challenge 3 - Connecting to the overall safety argument (P3)

The primary requirement in current work for assuring ML components are not related to the overall safety case. For example, the primary requirement in [3] is *'The residual risk associated with functional insufficiencies in the object detection function is acceptable'*. This is analogous to using reliability as a measure for safety. As a violation of these lower level requirements does not necessarily lead to unsafe behaviours, nor does meeting these requirements guarantee safety. It is therefore important to understand how the safety case for ML can be connected to the overall one, and how domain specific concerns can be traded-off to produce a safe system.

5 Implications

The statistic that that humans are the cause of 94% of road accidents [8] is often used as motivation for the adoption of autonomous vehicles; it is implied that the number of accidents would be reduced if the human driver was replaced with AI. Whilst there are *many* issues with this claim, what this data does not take in to account is all the accidents that human intervention prevented. By its nature this kind of data is difficult to model, however it is paramount that these subtle domain interactions are understood so that "good" safety criteria for ML algorithms can be established. This could be achieved through different ways through assurance process and outcome.

5.1 Change in Process and Outcome

In current assurance arguments, higher level safety requirements are decomposed into several lower level requirements with respect to properties such as hardware and software functionality. Therefore, it is proposed that decomposition of requirements through ML components should follow the same philosophy. However, the decomposition should occur with respect to domain-specific safety properties. This requires a deep understanding of the domain interactions, that must be skilfully mapped to the new operational context. For example, the *intent* of the heuristics that people use to avoid being on a designated AGV path should be incorporated into the SGV design. This presents a paradigm shift that goes well beyond the requirements specific only to the ML component, such as mitigating the effects of distributional shift.

5.2 Change in Relationship

Humans being assisted or replaced by systems that use AI necessitates a new way of thinking about trust and confidence that is different to traditional human-human assurance. The consideration of this area is outside the scope of this paper, however there has been significant advances to understand what is occurring inside the ML algorithm [5] which is likely to have a significant effect on the assurance process and outcome.

6 Conclusion

The nature of ML systems means that, whilst there is a strong consensus on many of the problems introduced, there is no unifying conceptualisation of the problem of assuring ML. This forms a barrier of communication between safety, ML developers, system engineers, *etc.*. In this paper, a new tripartite model of the challenge of assuring ML was presented to address this understanding issue. Using such a model for communication it is possible to co-ordinate inter-disciplinary work and improve both the quality and safety of the system.

Acknowledgements. Thanks to the Assuring Autonomy International Programme (AAIP) for support of this work.

References

1. Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., Mané, D.: Concrete problems in AI safety. arXiv preprint arXiv:1606.06565 (2016)
2. Banks, A., Ashmore, R.: Requirements assurance in machine learning. In: Proceedings of the AAAI Workshop on Artificial Intelligence Safety 2019. pp. 14–21. Springer (2018)
3. Burton, S., Gauerhof, L., Heinzemann, C.: Making the case for safety of machine learning in highly automated driving. In: International Conference on Computer Safety, Reliability, and Security. pp. 5–16. Springer (2017)
4. Douthwaite, M., Kelly, T.: Safety-critical software and safety-critical artificial intelligence: Integrating new practices and new safety concerns for ai systems. In: Proceedings of the Twenty-sixth Safety-Critical Systems Symposium (2018)
5. Gunning, D.: Explainable artificial intelligence (xai). Defense Advanced Research Projects Agency (DARPA), nd Web (2017)
6. Hawkins, R., Habli, I., Kelly, T.: The principles of software safety assurance. In: 31st International System Safety Conference (2013)
7. Koopman, P., Kane, A., Black, J.: Credible autonomy safety argumentation. 27th Safety-Critical Systems Symposium (February 2019)
8. NHTSA: National Motor Vehicle Crash Causation Survey: Report to Congress. Tech. rep., National Highway Traffic Safety Administration (July 2008)
9. Picardi, C., Hawkins, R., Paterson, C., Habli, I.: A pattern for arguing the assurance of machine learning in medical diagnosis systems. In: *(To appear)* International Conference on Computer Safety, Reliability, and Security (2019)
10. Rushby, J.: Quality measures and assurance for ai (artificial intelligence) software (1988)
11. Yampolskiy, R.: Taxonomy of pathways to dangerous ai. arXiv preprint arXiv:1511.03246 (2015)