Disruptive Innovations and Disruptive Assurance: Assuring Machine Learning and Autonomy

Robin Bloomfield, Adelard LLP and City University of London
Heidy Khlaaf, Philippa Ryan Conmy, and Gareth Fletcher, Adelard LLP

Autonomous and machine learning–based systems are disruptive innovations and thus require a corresponding disruptive assurance strategy. We offer an overview of a framework based on claims, arguments, and evidence aimed at addressing these systems and use it to identify specific gaps, challenges, and potential solutions.

Digital Object Identifier 10.1109/MC.2019.2914775
Date of publication: 27 August 2019

The advancement and adoption of machine-learning (ML) algorithms constitute a crucial innovative disruption. However, to benefit from these innovations within security and safety-critical domains, we need to be able to evaluate the risks and benefits of the technologies used; in particular, we need to assure ML-based and autonomous systems.

The assurance of complex software-based systems often relies on a standards-based justification. But in the case of autonomous systems, it is difficult to rely solely on this approach, given the lack of validated standards, policies, and guidance for such novel technologies. Other strategies, such as "driving to safety," that use evidence developed from trials and experience to support claims of safety in deployment are unlikely to be successful by themselves,[1,2] especially if the impact of security threats is taken into account. This

0018-9162/19©2019IEEE

reinforces the need for innovation in assurance and the development of an assurance methodology for autonomous systems.

Although forthcoming standards and guidelines will eventually have an important, yet indirect, role in helping us justify behaviors, we need further development of assurance frameworks that enable us to exploit disruptive technologies. In this article, we focus on directly investigating the desired behavior (e.g., the safety property or reliability) of a system through an argument- or outcome-based approach that integrates disparate sources of evidence, whether from compliance, experience, or product analysis. We argue that building trust and trustworthiness through argument-based mechanisms, specifically the claims, arguments, and evidence (CAE) framework (see "The Assurance Framework"), allows for the accelerated exploration of novel mechanisms that would lead to the quality advancement and assurance of disruptive technologies (see Figures S1 and S2 in the "The Assurance Framework" sidebar).

The key advantage of a claim-based approach is that there is considerable flexibility in how the claims are demonstrated since different types of arguments and evidence can be used as appropriate. Such a flexible approach is necessary when identifying gaps and challenges in uncharted territory, such as the assurance of ML-based systems. Indeed, CAE is commonly used in safety-critical industries (such as defense, nuclear, and medical) to assure a wide range of systems and devices and support innovation in assurance.

We are developing a particular set of CAE structures that is generically applicable and helps identify how to construct trustworthy ML-based systems by explicitly considering evidence of sources of doubt, vulnerabilities, and mitigations addressing the behavior of the system. In doing this, we not only assure and determine challenges and gaps in behavioral properties but also self-identify gaps within the assurance framework itself. In the remainder of this article, we describe our systematic approach to identifying a range of gaps and challenges regarding ML-based systems and their assurance.

IDENTIFYING ASSURANCE CHALLENGES

The decision to trust an engineering system resides in engineering argumentation that addresses the evaluation and risk assessment of the system and the role of the different subsystems and components in achieving trustworthiness. Although previous abstractions, models, and relationships have been constructed in CAE for the assurance of traditional software systems, it is not clear if the said existing blocks are sufficient to provide compositional argumentation enabling trustworthiness in ML-based systems. For example, domain-specific abstractions and arguments may need to be developed in CAE to specifically target ML subcomponents.

To develop a detailed understanding of such assurance challenges, we use CAE to create an outline of an overall assurance case, proceeding from top-level claims, concerning an experimental autonomous vehicle and its social context, down to claims regarding the evaluation of subsystems, such as the ML model (Figure 1). The case study autonomous vehicle, as is typical with similar state-of-the-art vehicles, contains a heterogeneous mixture of commercial off-the-shelf (COTS) components, including image recognition, lidar, and other items. Apportioning the trustworthiness, dependability, and requirements of each component to consider the real-time and safety-related nature of the system is challenging. In traditional safety-critical engineering, there would be diversity and defense in depth to reduce the trust needed in specific ML components; yet we do not know whether this is practicable for ML-based systems. Argumentation blocks may need to be further developed within CAE to determine how experimental data can allow for the comparison and assessment of diverse subsystems' contribution to defense in depth. This, in turn, can also inform future architectures of autonomous systems.

Beyond the study of the applicability of CAE to assure ML-based systems, the lens of the assurance case is used to identify gaps and challenges regarding techniques and evidence aimed at justifying desired system behaviors. This is further informed by a review of literature, a case study-based assessment of the experimental vehicle, and an investigation of our industry partners' development processes to assess the current state of the vehicle and the short- to medium-term future vision of its use case (approximately two years). To see how and whether security is addressed in the product lifecycle, we used the new U.K. Code of Practice PAS 11281, *Connected Automotive Ecosystems—Impact of Security on Safety*.[4]

In the subsequent sections, we discuss some of the gaps identified regarding technical capabilities that may enable trust of system behaviors. We highlight three areas: requirements, security, and verification and validation (V&V). There are also issues of ethics, advanced safety analysis techniques, defense in depth, and diversity modeling that we do not address.

GAPS AND CHALLENGES

Innovation, trust, and requirements

There is a need to address the realities of the innovation lifecycle and progressively

THE ASSURANCE FRAMEWORK

The claims, arguments, and evidence (CAE) framework supports the structured argumentation for complex engineering systems. It is based on an explicit claim-based approach to justification and relates back to earlier philosophical work by Wigmore[S6] and Toulmin[S7] as well as drawing on theory and empirical research in recent years in the safety and assurance cases areas (see John Rushby's analysis[S4] for a rigorous review of the field).

At the heart of the CAE framework are three key elements (Figure S1). Claims are assertions put forward for general acceptance. They are typically statements about a property of the system or some subsystem. Claims asserted as true without justification are assumptions, and claims supporting an argument are subclaims. Arguments link evidence to a claim, which can be deterministic, probabilistic, or qualitative. They consist of "statements indicating the general ways of arguing being applied in a particular case and implicitly relied on and whose trustworthiness is well established" (see Toulmin[S7]), together with validation of any scientific laws used. In an engineering context, arguments should be explicit. Evidence serves as the basis for justification of a claim. Sources of evidence can include the design, the development process, prior experience, testing (including statistical testing), or formal analysis.

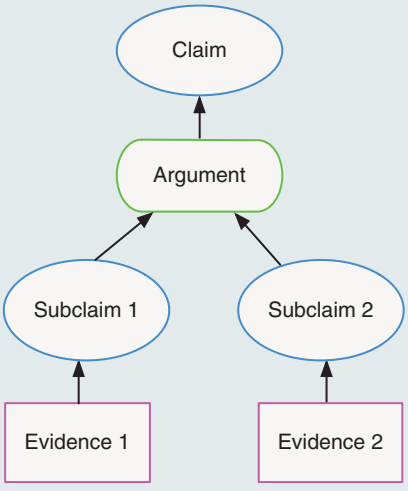

Figure S1. The CAE notation.

In addition to the basic CAE concepts, the framework consists of CAE blocks that provide a restrictive set of common argument fragments and a mechanism for separating inductive and deductive aspects of the argumentation (Figure S2). These were identified by empirical analysis of actual safety cases.[S5] The blocks are as follows:

- *Decomposition*: There is partition of some aspect of the claim, or divide and conquer.
- *Substitution*: A claim about an object is refined into a claim about an equivalent object.

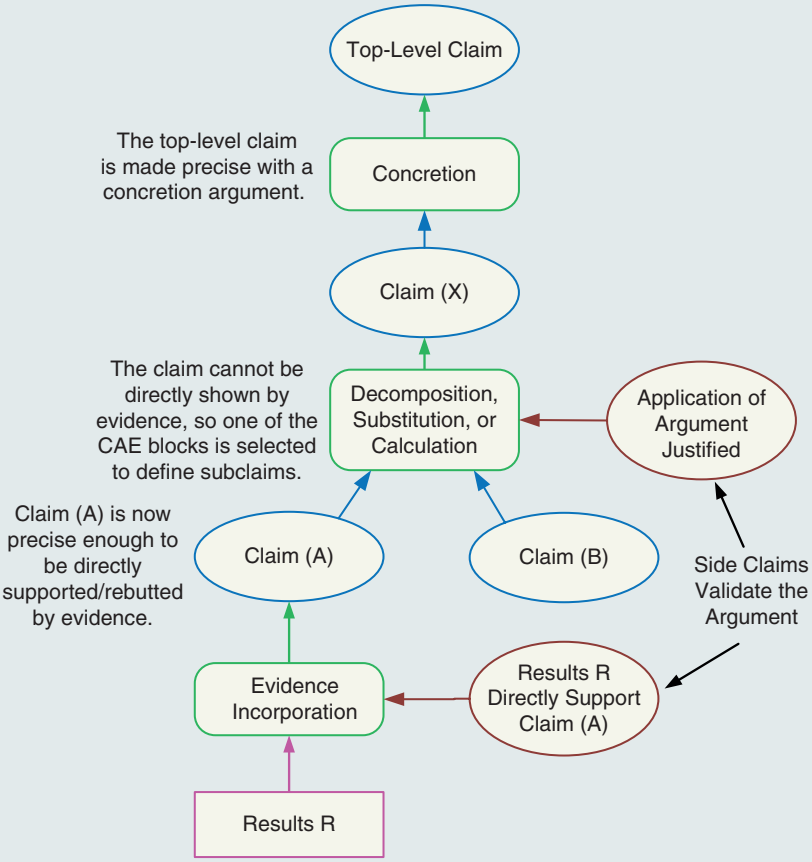

Figure S2. An example of CAE block use.

(*continued*)

THE ASSURANCE FRAMEWORK (Cont.)

- *Evidence incorporation*: Evidence supports the claim, with an emphasis on direct support.
- *Concretion*: Some aspect of the claim is given a more precise definition.
- *Calculation or proof*: Some value of the claim can be computed or proved.

The framework also defines connection rules to restrict the topology of CAE graphical structures. The use of blocks and associated narrative can capture challenges, doubts, and rebuttals and illustrates how confidence can be considered as an integral part of the justification.

The basic concepts of CAE are supported by an international standard,[S1] IAEA guidance,[S3] and industry guidance.[S2] To support CAE, a graphical notation can be used to describe the interrelationship of evidence, arguments, and claims.[S3,S5] In practice, top desirable claims, such as "the system is adequately secure," are too vague or are not directly supported or refuted by evidence. Therefore, it is necessary to create subclaim nodes until the final nodes of the assessment can be directly supported or refuted by evidence.

REFERENCES

S1. *Systems and Software Engineering—Systems and Software Assurance, Part 2: Assurance Case*, ISO/IEC 15026-2:2011, 2011.

S2. P. G. Bishop and R. E. Bloomfield, "A methodology for safety case development," in *Industrial Perspectives of Safety-Critical Systems: Proceedings of the Sixth Safety-Critical Systems Symposium, Birmingham 1998*, F. Redmill and T. Anderson, Eds. London: Springer-Verlag, 1998, pp. 194–203.

S3. International Atomic Energy Agency, "Dependability assessment of software for safety instrumentation and control systems at nuclear power plants," IAEA Nuclear Energy Series NP-T-3.27, 2018. [Online]. Available: https://www-pub.iaea.org/books/IAEABooks/12232/Dependability-Assessment-of-Software-for-Safety-Instrumentation-and-Control-Systems-at-Nuclear-Power-Plants

S4. J. Rushby, "The interpretation and evaluation of assurance cases," SRI Int., Menlo Park CA, Tech. Rep. SRI-CSL-15-01, July 2015.

S5. R. Bloomfield and K. Netkachova, "Building blocks for assurance cases," in *Proc. IEEE Int. Symp. Software Reliability Engineering Workshops (ISSREW)*, Nov. 2014, pp. 186–191. doi: 10.1109/ISSREW.2014.72.

S6. J. H. Wigmore, "The science of judicial proof," *Virginia Law Rev.*, vol. 25, no. 1, pp. 120–127, Nov. 1938. doi: 10.2307/1068138.

S7. S. E. Toulmin, *The Uses of Argument*. Cambridge Univ. Press, United Kingdom. 1958.

develop requirements, including those for trustworthiness and assurance. In this innovation approach, the vehicle is gradually developed from a platform to trial technologies to the final product (Figure 2). There is an assurance gap in that, when analyzing how much the technologies need to be trusted, there must be an articulated vision of what they will be used for. If the vision of how something will be used is not clearly formulated, we cannot assess how much we need to trust it or what the risks are.

This is particularly important for security and systemic risks, where the scale and nature of the deployment (such as a key part of an urban transport system) will lead to more onerous requirements that have to be reflected in the earlier technology trials and evaluations. Alternatively, more agile approaches would be to progressively identify these trust requirements as the innovation proceeds. But this might lead to solutions that do not scale and, in the extreme, could not be deployed. We believe that the innovation lifecycle subsequently presented is typical for many players in the industry and will be increasingly adopted as the ML components become more productized.

Security

Security is a fundamental and integral attribute of the technical themes of the project, in the requirements, V&V, and assurance research. While the requirements of the new PAS 11281 Code of Practice may be met in a mature implementation of the vehicle being studied, on the whole, the security will be challenging for industry, and advice must be provided on partial and project-specific implementation of the PAS that allows for maturity growth.

The security aspects need to be integrated into the entire lifecycle: systems are not safe if they are not secure. This applies to the vehicle as a whole as well as to the ML subsystems; most ML systems have not been designed with a systematic attention to security.[10] The PAS clauses address the following areas and are equally applicable to the vehicle and its components:

1. security policy, organization, and culture
2. security-aware development process
3. maintaining effective defenses
4. incident management
5. secure and safe design
6. contributing to a safe and secure world.

As we noted previously, the deployment of autonomous technologies may follow an innovation lifecycle that first focuses on functionality and seeks to progressively add additional assurance and security. This will make

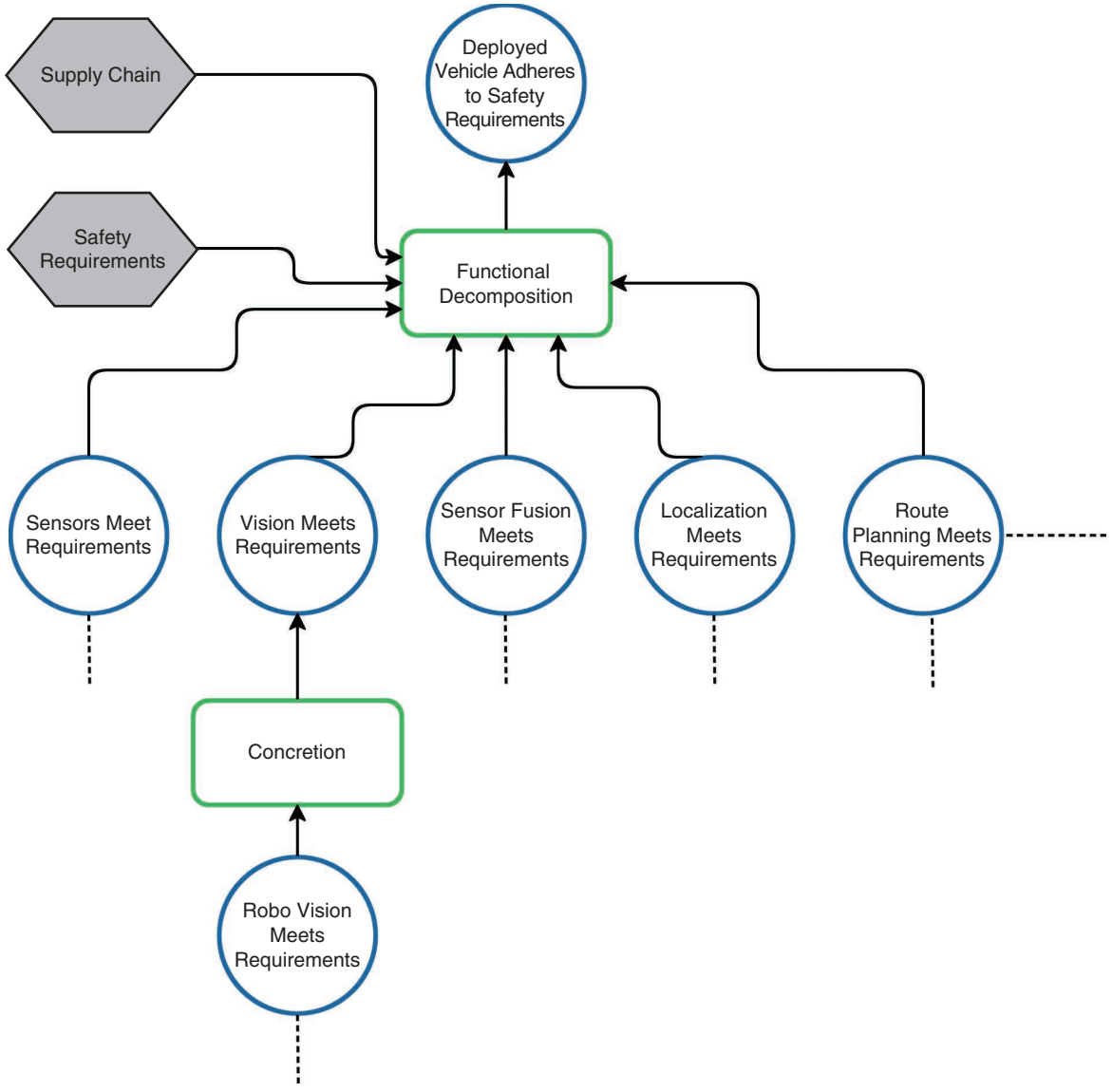

Figure 1. A high-level example of an assurance subcase in CAE.

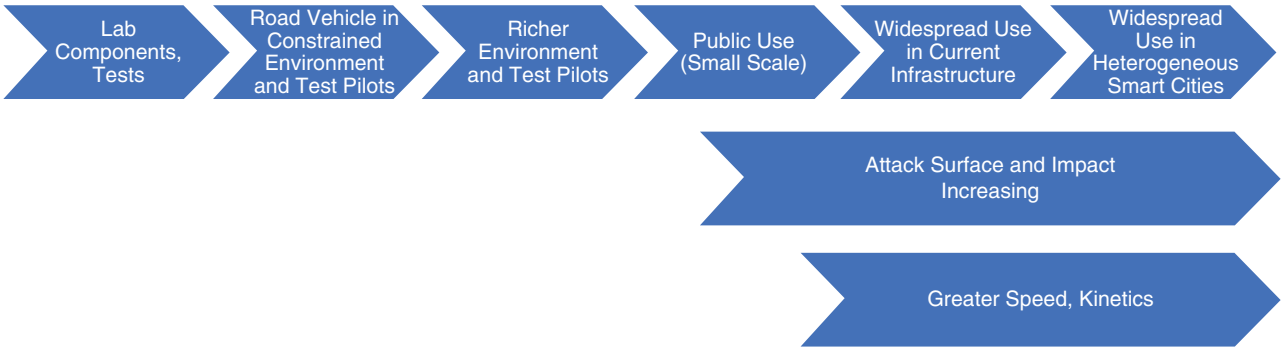

Figure 2. The typical stages of development from innovation to products.

the development of the assurance and safety cases and associated security and safety risk assessments particularly challenging. From our experience, we recommend the following:

1. Explicitly define the innovation cycle and assess the impact and feasibility of adding assurance and security.
2. Address the approach to security-informed safety at all stages of the innovation cycle. If safety, security, and resilience requirements are largely undefined at the start of the innovation cycle, the feasibility of progressively identifying them during the cycle should be assessed, together with the issues involved in evolving the architecture and increasing the assurance evidence.
3. Apply PAS 11281 to systematically identify the issues. Use a CAE assurance case framework and map PAS clauses to this to provide a systematic approach to applying the PAS.
4. Consider a partial and project-specific implementation of the PAS to meet the innovation cycle.
5. Collect experience in developing a security-informed safety case and integrating security issues into the safety analyses needed to implement the PAS.

V&V

We use the assurance case in CAE top-down to identify the claims we wish to support and bottom-up to evaluate the evidence that could be provided by them and, hence, systematically assess gaps, challenges, and solutions. This is shown schematically in Figure 3. As part of this analysis, we assessed

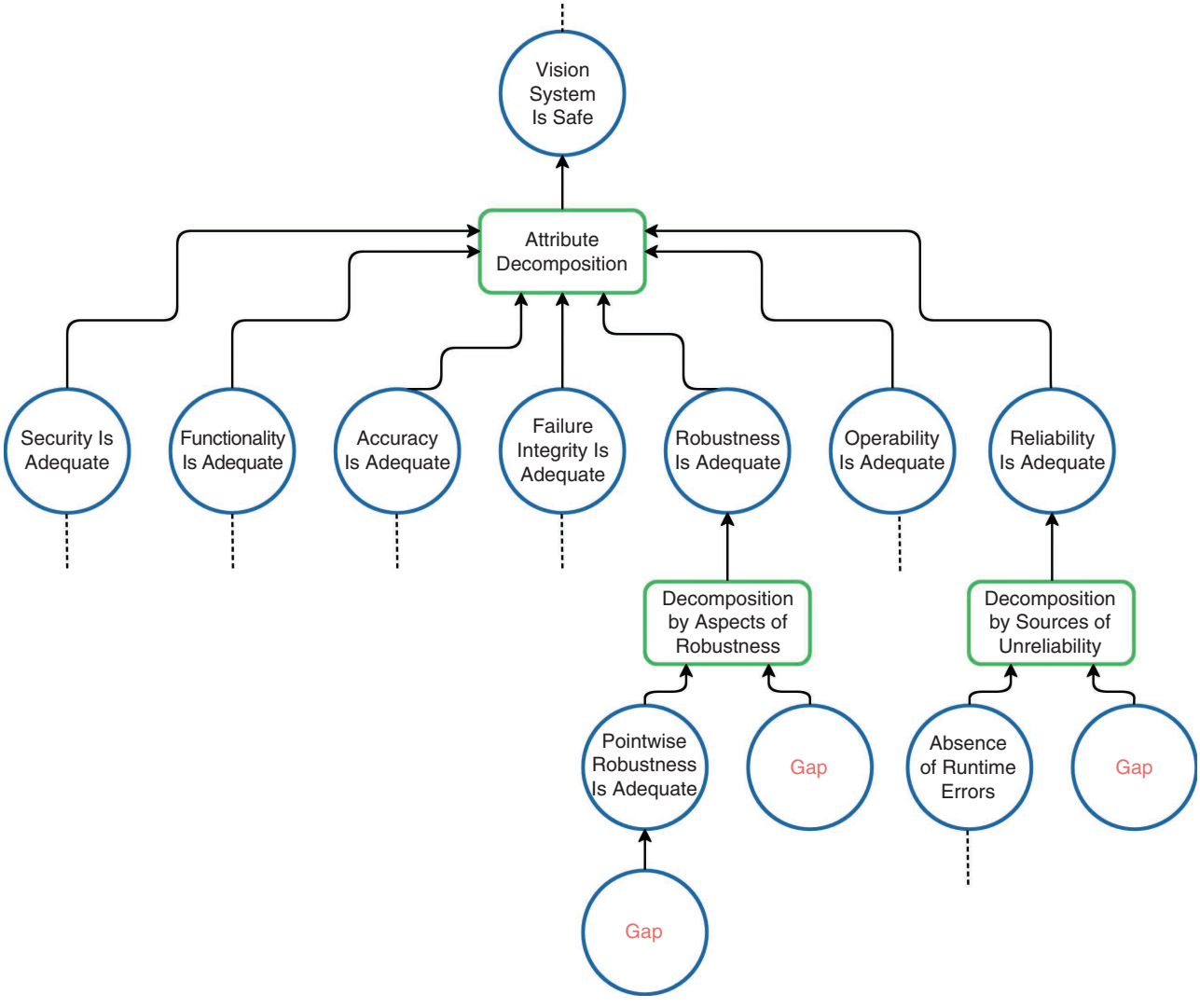

Figure 3. The use of CAE to assess V&V gaps.

state-of-the-art formal methods for autonomous systems and observed that their maturity and applicability are lacking for sufficiently justifying behavioral and vulnerability claims.

Consider the issue of adversarial attacks and perturbations,[5,6] which has been particularly challenging with regard to the robustness of ML algorithms. Verification researchers have focused on the property of pointwise robustness, in which a classifier function f' is not robust at point x if there exists a point y within η such that the classification of y is not the same as the classification of x. That is, for some point x from the input, the classification label remains constant within the neighborhood η of x, even when small-value deltas (i.e., perturbations) are applied to x. A point x would not be robust if it were at a decision boundary, and adding a perturbation would cause it to be categorized in the next class. Generally speaking, the idea is that a neighborhood η should be reasonably classified as the given class.

However, proposed pointwise robustness verification methods[8–10] suffer from the same set of limitations.

- There is a lack of clarity on how to define meaningful regions η and manipulations.
  - The neighborhoods surrounding a point x that are currently used are arbitrary and conservative.
- We cannot enumerate all x points near which the classifier should be approximately constant; that is, we cannot predict all future inputs.

Furthermore, researchers have been unable to find compelling threat models that required perturbation indistinguishability,[12] and it has been demonstrated that l_p, which defines the neighborhood region η, is a poor proximity for measuring what humans actually see.[13] Finally, adversarial perturbations can be achieved by much simpler attacks that do not require ML algorithms (e.g., covering a stop sign). Thus, the extent to which these techniques can provide us with any level of confidence is not very high.

Other verification techniques[7,9] aim to verify more general behaviors regarding ML algorithms, instead of just pointwise robustness. Such techniques require functional specifications, written as constraints, to be fed into a specialized linear-programming solver to be verified against a piecewise linear constraint model of the ML algorithm. However, the generalization of these algorithms is challenging, given the requirement of well-defined and bounded traditional system specifications, devoid of specifications regarding the behavior of the ML algorithm itself. These techniques are thus applicable to well-specified deterministic ML algorithms and cannot be applied to perception algorithms, which are notoriously difficult to specify, let alone verify.

> It is unclear how potential faults arising from dynamic languages could affect the functionality of an ML model itself.

Apart from the ML algorithm, the assurance of the non-ML supporting components of an autonomous system is challenging, given that the use of COTS or open source components leads to uncertain provenance. Errors within non-ML components can propagate and affect the functionality of the ML model.[14] It is, therefore, important to explore how traditional V&V methods—in particular, static analysis of C code—can provide assurance for the larger ML system, offering confidence beyond the component level. In the following, we provide a preliminary list of results from analyzing YOLO, a commonly used open source ML vision software, and a number of different run-time errors that were identified:

- a number of memory leaks, such as files opened and not closed, and temporarily allocated data not freed, leading to unpredictable behavior, crashes, and corrupted data
- a large number of calls to free where the validity of the returned data is not checked [this could lead to incorrect (but potentially plausible) weights being loaded to the network]
- potential "divide by zeros" in the training code (this could lead to crashes during online training, if the system were to be used in such a way)
- potential floating-point divide by zeros, some of which were located in the network cost calculation function (as noted above, this could be an issue during online training).

These errors would be applicable only to languages such as C and C++. Not all errors would be relevant to a language such as Python, used in the implementation of numerous ML libraries and frameworks, as the semantics and implementation of the language itself do not enable overflow/underflow errors, defined by Hutchison et al.[14] However, Python is a dynamically typed language, bringing about a different set of program errors not exhibited by statically typed languages (such as type errors). Unfortunately, no static analysis techniques or tools exist to allow for the analysis of Python code. Furthermore, it is unclear how potential faults arising from dynamic languages could affect the functionality of an ML model itself. This is a large gap within the formal methods field that needs to be addressed immediately, given the deployment of autonomous vehicles utilizing Python.

There is a need for disruptive innovation in the assurance of autonomous and ML-based systems. We provided a summary of the outcome-focused, CAE-based framework we are evolving to address these systems and used it to identify specific gaps and challenges; we also discussed some solutions. We demonstrated the feasibility of deploying the best of existing work (e.g., advanced static analysis techniques) and identified the need for new approaches.

Overall, there is a need for stronger evidence and techniques to assure the dependability of ML components and for autonomous systems as a whole. Indeed, there is common good in sharing techniques and strategies regarding development lifecycles, diversity, security, and V&V algorithms in sufficient detail for independent analysis and research. We hope to play our part in this by sharing our generic developed assurance case and providing, in the public domain, the more detailed report this article is based on. If we can achieve our goal of disruptive assurance, this can have a positive impact on innovation in a wide range of industries and technologies, not just ML-based ones.

ACKNOWLEDGMENTS

This article discusses work undertaken within the Towards Identifying and closing Gaps in Assurance of autonomous Road vehicleS (TIGARS) project. The project is a collaboration between Adelard, Witz, the City University of London, the University of Nagoya, and Kanagawa University. This work is partially supported by the Assuring Autonomy International Programme, a partnership between Lloyd's Register Foundation and the University of York. We acknowledge the additional support of the U.K. Department for Transport.

REFERENCES

1. N. Kalra and S. Paddock, *Driving to Safety: How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability*? Santa Monica, CA: RAND Corporation, 2016. [Online]. Available: https://www.rand.org/pubs/research_reports/RR1478.html
2. P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE Int. J. Transp. Safety*, vol. 4, no. 1, pp. 15–24, 2016.
3. R. Bloomfield, P. Bishop, E. Butler, and R. Stroud, "Security-informed safety: Supporting stakeholders with codes of practice [Cybertrust]," *Computer*, vol. 51, no. 8, pp. 60–65, Aug. 2018.
4. *Connected Automotive Ecosystems—Impact of Security on Safety*, British Standards Institution, PAS 11281, 2018.
5. C. Szegedy et al., Intriguing properties of neural networks. 2013. [Online]. Available: https://arxiv.org/abs/1312.6199
6. I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learning Representations—Computational and Biological Learning Society*, 2015. [Online]. Available: https://arxiv.org/abs/1412.6572v3
7. L. Pulina and A. Tacchella, "An abstraction-refinement approach to verification of artificial neural networks," Computer Aided Verification, CAV 2010, Lecture Notes in Computer Science, vol 6174, T. Touili, B. Cook, and P. Jackson, Eds. Berlin: Springer, pp. 243–257.
8. X. Huang, M. Kwiatkowska, S. Wang, and M. Wu, Safety verification of deep neural networks. 2016. [Online]. Available: https://arxiv.org/abs/1610.06940
9. G. Katz, C. Barrett, D. Dill, K. Julian, and M. Kochenderfer, Reluplex: An efficient SMT solver for verifying deep neural networks. 2017. [Online]. Available: https://arxiv.org/abs/1702.01135
10. N. Papernot, P. McDaniel, A. Sinha, and M. Wellman, Towards the science of security and privacy in machine learning. 2016. [Online]. Available: https://arxiv.org/abs/1611.03814
11. W. Ruan, X. Huang, and M. Z. Kwiatkowska, Reachability analysis of deep neural networks with provable guarantees. 2018. [Online]. Available: http://arxiv.org/abs/1805.02242
12. J. Gilmer, R. Adams, I. Goodfellow, D. Andersen, and G. Dahl, Motivating the rules of the game for adversarial example research. 2018. [Online]. Available: https://arxiv.org/abs/1807.06732
13. Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? A new look at signal fidelity measures," *IEEE Signal Process. Mag.*, vol. 26, no. 1, pp. 98–117, 2009.
14. C. Hutchison et al., "Robustness testing of autonomy software," in *Proc. IEEE/ACM 40th Int. Conf. Software Engineering: Software Engineering in Practice Track*, Gothenburg, Sweden, May 27–June 3, 2018, pp. 276–285.

ROBIN BLOOMFIELD is with Adelard LLP and the City University of London. Contact him at reb@adelard.com or reb@csr.city.ac.uk.

HEIDY KHLAAF is with Adelard LLP. Contact her at hak@adelard.com.

PHILIPPA RYAN CONMY is with Adelard LLP. Contact her at pmrc@addelard.com.

GARETH FLETCHER is with Adelard LLP. Contact him at gtf@adelard.com.