What Is Acceptably Safe for Reinforcement Learning?

John Bragg[1](✉) and Ibrahim Habli[2]

[1] MBDA UK Ltd., Filton, Bristol, UK
john.bragg@mbda.co.uk
[2] University of York, York, Yorkshire, UK
ibrahim.habli@york.ac.uk

Abstract. Machine Learning algorithms are becoming more prevalent in critical systems where dynamic decision making and efficiency are the goal. As is the case for complex and safety-critical systems, where certain failures can lead to harm, we must proactively consider the safety assurance of such systems that use Machine Learning. In this paper we explore the implications of the use of Reinforcement Learning in particular, considering the potential benefits that it could bring to safety-critical systems, and our ability to provide assurances on the safety of systems incorporating such technology. We propose a high-level argument that could be used as the basis of a safety case for Reinforcement Learning systems, where the selection of 'reward' and 'cost' mechanisms would have a critical effect on the outcome of decisions made. We conclude with fundamental challenges that will need to be addressed to give the confidence necessary for deploying Reinforcement Learning within safety-critical applications.

Keywords: Safety · Assurance · Artificial Intelligence
Machine Learning · Reinforcement Learning

1 Introduction

Until recently, the safety assurance of software systems has been a reasonably well-established activity where safety engineers are able to provide evidence and confidence that the software will behave in a predictable way, for a given set of inputs and defined operating environment. However, the increasing use of Artificial Intelligence (AI) and Machine Learning (ML) has moved the goalposts; we can no longer rely on the 'traditional' safety techniques that depend on the 'deterministic' nature of the software [1]. The program that runs tomorrow may make different decisions and choices to the program that is executing today; it will have 'learned' from its experience.

AI and ML systems are essentially software systems that learn and adapt, based on experience and their environment [2]. Software is the primary vehicle for the implementation of AI and ML. However, historically, industry has typically

B. Gallina et al. (Eds.): SAFECOMP 2018 Workshops, LNCS 11094, pp. 418–430, 2018.
https://doi.org/10.1007/978-3-319-99229-7_35

shied away from putting safety-critical functionality into complex software that implement ML, e.g. preferring hardware technologies as they are relatively easier to provide assurance from a safety perspective. This position is not sustainable for future robotics and autonomous systems that are expected to learn and adapt yet maintain 'safety' [3]. Unfortunately, *"safety engineering is lagging behind emergent technologies."* [4] Although advances in safety analysis and assurance are far behind, the safety community may be able to catch-up but realistically may not be able to get ahead of the curve. The ML technology will almost always outpace safety assurance. As such, if a solution cannot be shown to be safe it may never become acceptable to deploy in safety-critical applications, despite the benefits that it may bring.

In this paper we explore the implications of the use of ML in safety-critical applications. We focus on Reinforcement Learning (RL), where the selection of 'reward' and 'cost' mechanisms would have a critical effect on the safety outcome of the decisions made. We propose a high-level argument that could be used as the basis of a safety case for RL systems and identify and analyse the technical and socio-technical factors that affect the potential strength of the reasoning. We conclude with fundamental challenges that will need to be addressed to give the confidence necessary for deploying RL within safety-critical applications.

2 Machine Learning

ML is a technique that relies on algorithms to analyse huge datasets and allows a machine (e.g. a computer) to perform predictive analysis, faster than any human is able to do. ML works, not by programming the computer to undertake a specific task, but by programming a computer with algorithms that provide it with the ability to learn how to perform a task. Whilst the implementation of ML does not in itself make a system an AI, it enables the system to perform the following tasks:

- Adapt to new circumstances that the original developer or system designer didn't envision;
- Detect patterns in different types of data sources;
- Create new behaviours based on recognised patterns; and
- Make decisions based on the success or failure of these behaviours.

The utilisation of ML is becoming more prevalent and is spreading to multiple domains as it is a feasible and cost-effective solution for many tasks that would be otherwise prohibitive to accomplish by any other means.

2.1 Reinforcement Learning

Figure 1 shows a model of how Reinforcement Learning (RL) works. An agent will be in a given state, it will perform an action on its environment and some time later will receive a response from that environment relating to the outcome

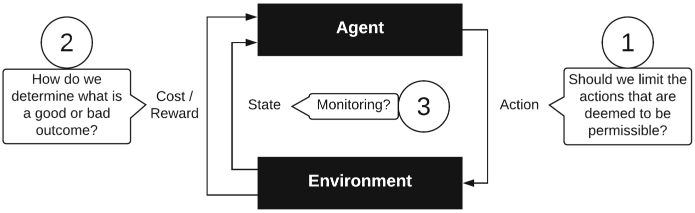
Agent
Environment
2
How do we determine what is a good or bad outcome?
Cost / Reward
State
Monitoring?
3
Action
1
Should we limit the actions that are deemed to be permissible?

Fig. 1. A simple model of Reinforcement Learning

of that action. The agent will have 'learnt' something from that course of action and may subsequently change state. The cycle then repeats.

Whilst this is a simplified view of how RL works, it still raises some interesting issues:

1. Are we able to determine what the permissible actions are? If we do this, are we going to unnecessarily constrain the agent in carrying out what might actually be the correct (safe) course of action?
2. The agent is going to learn both 'good' and 'bad' as a result of its actions (negative outcomes can be as good as positive ones from the learning perspective).
3. Should we monitor the system and 'fail safe' if there are indications that it is moving towards an unsafe state? If so, to what extent can we understand and explain the internal mechanisms by which decisions are made by the RL System?

The concept of the reward mechanism is key to the way in which this type of system will learn, with the potential for such systems to make 'very bad' decisions in terms of the resulting outcome; in the case of safety-related/critical systems we are of course talking about the concept of 'harm'.

3 Safety of Complex Engineering Systems

When considering the safety of a system we are fundamentally referring to hazards, risks, and accidents. A hazard can be thought of as 'a potential source of harm'. This could be harm to people, assets or the environment in which the system is operating. An accident is the harm that occurs due to a hazard not being sufficiently controlled. A simple example of which would be the hazard of high pressure gas being contained within a canister. An accident would be violent rupture of the canister due to over-pressurisation. The risk that this accident occurs can be mitigated by manufacturing the canister from a material strong enough to be able to deal with pressures in excess of the expected operating pressure. The risk can be further reduced by adding other mitigations, such as a pressure release valve, that will activate before damage to the canister can occur.

System safety is an attribute of the configuration of the System components (i.e. sub-systems) the environment(s) within which the System exists or operates, and the hazards and accident outcomes associated with the System. In order to start to address System Safety it is necessary to understand how the functionality and the potential hazardous behaviours of the overall system, each System component, the relationships and interactions between them and the influences of the environment(s), are defined and studied.

3.1 Safety-I: The 'Traditional' Approach

A deterministic system (or perhaps more accurately a 'predictable system'), which is the focus of most safety standards, is one that will produce the same output from a given set of input parameters, starting conditions, or initial state [1]. The approach to safety in these systems is to construct an argument around the assumptions made during the design phase and the safety constraints based on those assumptions, that are subsequently placed upon it to ensure 'safe' operation. Accordingly, the predictions made about the behaviour of the system and the interactions that the system will have with its environment (including other systems, users and processes) will drive the safety argument structure. Evidence provided in support of the safety case is gathered through the application of analyses such as: Fault Tree Analysis (FTA), Failure Modes and Effects Analysis (FMEA) and Software Hazard Analysis and Resolution in Design (SHARD) [5].

As discussed by Hollnagel et al. in [6,7], historical approaches to safety presume *"...things go wrong because of identifiable failures or malfunctions of specific components"*. The safety of a system therefore depends on the enforcement of constraints on the behaviour of the system features, including constraints on their interactions, in order to avoid failure. The safety case presents a safety argument to this effect and evidence is provided to support it. This type of safety reasoning is defined as Safety-I.

Whether the safety case for the system remains valid for the operational life of the system depends upon whether those predictions continue to hold true. If they do not, then the confidence in the safety argument can diminish and the validity of the safety assurance case is then challenged. The safety argument must then be re-visited.

3.2 The Limitations of Safety-I

For inherently complex systems, such as those using ML, the future state of the system cannot be predicted. It is difficult to foresee all potential failure modes, and thus the implementation of system constraints during system design are not able to assure the through-life safety of the system. As stated in [8]:

> *"...the learning, emergent and adaptive behaviour and use of these systems pose a significant challenge to the assumptions and predictions made in the safety case, regardless of whether the documentation of the safety argument is explicit (i.e. in goal-based certification) or implicit (i.e. in prescriptive certification)."*

Thus, 'traditional' safety methods have limited use: firstly because any type of ML must almost certainly be implemented within software and the 'traditional' methods are predominantly focused on hardware; secondly, these methods tend to be used pre-deployment and are not typically used post-deployment to monitor safety functions.

3.3 Safety-II: The 'Adaptive' Approach to Safety

The introduction of the Safety-II approach brings a fundamental paradigm change to system safety; moving from the Safety-I approach of ensuring that *"as few things as possible go wrong"* to ensuring that *"as many things as possible go right."* [7] Safety-II relates to the system's ability to adapt and succeed under varying conditions. The safety of the system is therefore ensured, not through the implementation of pre-defined constraints but through its ability to deal with situations that cannot be predicted [7] and applying adjustments to the system during its operational life. The safety analysis of the system would thus be predicated on the ability of the system to avoid hazardous conditions or adapt and deal with them should they occur.

RL Systems adapt their behaviour based on the decisions they have made and the response they receive from their environment. This is similar to how a human being deals with day-to-day situations, i.e. making adjustments and trade-offs based on understanding their actual, and often dynamic, environment.

From a Safety-II perspective, we need an approach that aligns itself with this type of 'dynamism'. In their paper [8], Denney et al. identify three principles for dynamic safety cases that have to:

1. Pro-actively compute the confidence in, and update the reasoning about, the safety of ongoing operations;
2. Provide an increased level of formality in the safety infrastructure; and
3. Provide a mixed-automation framework.

The above safety assurance challenges, in particular with respects to RL Systems, are explored and refined in the next section.

4 Reinforcement Learning and Safety

In Fig. 2 we propose a high-level safety argument for RL, represented using the Goal Structuring Notation (GSN) [9]. GSN is a widely-used graphical language for formulating the structure of a safety argument. It provides a concise way of describing the claims of the argument (represented by goals drawn as rectangular shapes), the context that is applicable to the claim being made (represented by the rounded rectangle shapes), and the strategies that are used to explain the link between a goal and one or more sub-goals (represented by the parallelogram shapes). The diamond shapes represent an undeveloped entity, i.e. one where further refinement will be required to substantiate the claim made.

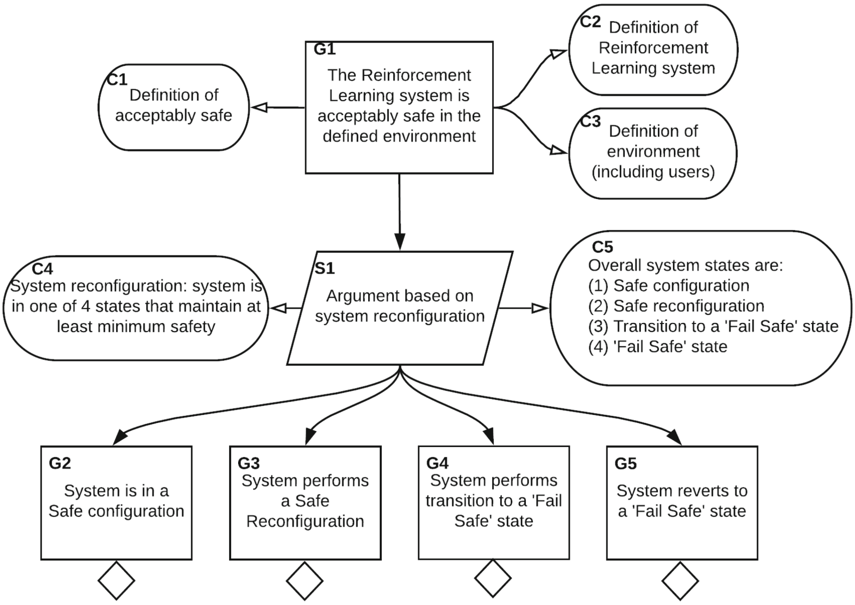
G1
The Reinforcement Learning system is acceptably safe in the defined environment
C1
Definition of acceptably safe
C2
Definition of Reinforcement Learning system
C3
Definition of environment (including users)
S1
Argument based on system reconfiguration
C4
System reconfiguration: system is in one of 4 states that maintain at least minimum safety
C5
Overall system states are:
(1) Safe configuration
(2) Safe reconfiguration
(3) Transition to a 'Fail Safe' state
(4) 'Fail Safe' state
G2
System is in a Safe configuration
G3
System performs a Safe Reconfiguration
G4
System performs transition to a 'Fail Safe' state
G5
System reverts to a 'Fail Safe' state

Fig. 2. A high-level safety argument structure for Reinforcement Learning systems

4.1 Argument Structure

The argument shown in Fig. 2 starts with a top-level goal, G1, that the RL System is ‘acceptably safe’; this is a typical top-level claim when constructing a safety argument [10]. This claim is linked to a number of nodes, which define the context of the claim. The terms ‘acceptably safe’ and ‘system’ need to be defined, but of particular interest to the argument is the ‘environment’. As previously discussed, the environment is the main point of interaction for the RL System and will define the nature of the ‘Reward’ or ‘Cost’ that the system receives.

Our main strategy for arguing the safety of a RL System is to provide an ‘Argument based on system reconfiguration’ (node S1). That is, the system is operating in a ‘safe’ configuration, in a ‘Fail Safe’ state, in a safe reconfiguration or transition state, all of which should guarantee at least minimum safety, despite a potential reduction in functionality or availability. This covers a number of sub-claims (nodes G2 through to G5). Firstly we have a sub-claim (G2) that the ‘System is in a Safe configuration’, this is represented as an undeveloped goal but is likely to be supported by verification and validation evidence that this is the case (including run-time verification evidence). Node G3 represents the claim that the RL System is able to perform a ‘Safe reconfiguration’, meaning that in response to a reward or punishment as part of its learning cycle the system can reconfigure its behaviour ‘safely’, i.e. it does not introduce any new hazardous behaviour.

If a ‘safe’ reconfiguration is not possible, assurance is needed that the system is instead able to transition ‘safely’ (node G4) into a ‘Fail Safe’ state (node G5). The evidence to support nodes G3, G4 and G5 in terms of verification and validation will be required (though this is beyond the scope of this paper, which focuses on the argument).

Table 1. Challenges with Reinforcement Learning

Challenge area	Short description
Societal Acceptance	How do we choose which human and social values to embody within the system? Sect. 4.2
Risk vs. Benefits	Do the benefits of deploying a RL System outweigh the safety risks? Is this a sufficient threshold? Sect. 4.3
Cost vs. Reward	How does a RL System learn where a negative outcome still needs to be ‘safe’? Sect. 4.4
Monitoring & Feedback	What mechanisms should be used for an RL System to decide that it needs to reconfigure? Sect. 4.5
Learning Constraints	Can safety only be achieved through constraining the way in which a RL System learns? Sect. 4.6
Fail Safe	How and when should RL Systems fail safe? Sect. 4.7
Intelligent Safety	How can a RL System update its own safety case and explain the choices it has made? Sect. 4.8

Table 1 captures a number of technical and socio-technical challenge areas that we believe are key to the aim of assuring RL, and therefore in supporting the argument presented. These areas are discussed in the next sections and are linked to the nodes in the GSN argument.

4.2 Societal Acceptance

Ethics, safety and societal impact are three emotive topics that can be challenging enough in non-AI systems [11]. Factor in the current love/hate relationship that the public has with ML, as well as the ‘unpredictable’ nature of AI systems, and these sensitivities become even more entrenched. These are areas that Google’s DeepMind are looking to address, where DeepMind’s Mustafa Suleyman believes that the ethics of ML must be central to its development [12].

In [13], Stuart Russell proposed *“3 Principles for Safer Artificial Intelligence”*, which were:

Altruism: the robot’s only objective is to maximise the realisation of human values;

Humility: the robot is initially uncertain about what those values are;

Observation of human choices: human behaviour profiles information about human values.

However, Russell's proposed principles raise their own questions. If the aim of AI is to replace or augment human decision making and maximise the realisation of human values, then we have to ask the question 'whose human values do we choose?' Not everyone's values are the same, especially in cases involving trade-offs between different values including safety, privacy and liberty. We have different belief systems, cultures, morals and ethics. How do we choose which ones to embody within a system?

The study of machine ethics is important when considering safety (Bostrom et al. [14] and Dennis et al. [15] are just two examples) but is the subject of machine ethics a red herring? It could be. As stated in Yampolskiy in 2012 [16], we need to ensure that machines are inherently safe and law abiding. The application of ethics to a machine is fraught with issues and could ultimately cause dissension. So, should ML systems exploit their computational abilities rather than emulate any such flawed human behaviour?

Considerations on the above topics would provide support for nodes C3 (Who are our users and the wider stakeholders and what are their values?) and C1 (What is acceptable to them?) in Fig. 2. This is key in defining acceptable risks, potential costs and intended benefits and a fair distribution of the risks, costs and benefits between the system stakeholders.

4.3 Risk vs. Benefits

In Sect. 3 we discussed that engineering a safe system always comes with a certain amount of risk [17,18]. However, when we introduce a RL System the challenge of managing uncertainty (and thus risk) can significantly increase given that we are unable to ascertain, prior to deployment, that the RL algorithm will make a 'safe' decision. The algorithm will make a decision based on a model calculated from its input data and this therefore comes with a probability that the decision made is a wrong one, especially during the early operation of the system.

However, not all problems are safety-related, and a wrong decision may not necessarily lead to harm. Even if, for example, the data that the RL System is using is corrupted it is not a foregone conclusion that an accident is going to occur. It may still be safe to deploy RL even in these error conditions (incorrect outcomes are not always unsafe outcomes). Indeed, in relation to the previous section where we discussed the impact on humans, we know from experience and evidence that humans often struggle to make decisions about safety [19], and therefore the distribution of risks and trade-offs with a RL System may well be an improvement for the majority of situations. Additional benefit may come, not only by ensuring that hazards are avoided, but in actually reducing the risk beyond that which was previously envisioned because the system has 'learnt' how to do this, i.e. in ways that a human could not have predicted.

These trade-offs would need to be explicitly considered as part of the safety argument and would largely contribute to nodes G2 and G3 in Fig. 2. That is, to what extent can we trust the RL System to reconfigure and maintain, or even improve, safety before forcing it to transition to a constrained 'Safe state'?

In other words, under certain conditions, excessive safety constraints might reduce rather than improve overall safety.

4.4 Cost vs. Reward

As we have discussed, RL is based on a cost/reward feedback mechanism. If the system makes a decision that is good then it is rewarded, if the decision is bad then there is a cost (or punishment) associated with that decision or action. The challenge for safety-critical applications is to select a cost/reward mechanism that is appropriate to the nature of the system, its objectives, and the safety constraints that need to be adhered to [20]. This is also linked to the risks and benefits discussed in Sect. 4.3, whereby choosing a cost/reward mechanism that is 'overly-protective' could result in limited benefits of using RL in the first place. Conversely, selecting a cost/reward mechanism that is too liberal could result in a significant increase in risk.

Within the context of a Safety-II approach, 'Cost' and 'Reward' are closely linked to risks and benefits. Therefore, considerations on cost (including increased safety risk) and reward (including increased safety benefit) support the claims being made in nodes G2 and G3 in Fig. 2. For example, a potential configuration might increase the safety margin (i.e. reward) but might compromise certain privacy aspects (i.e. cost). How could the RL System decide if the reward would outweigh the cost in this configuration, if this would be an acceptable trade-off by those affected by the decision and if, who, how and when to consult human stakeholders?

4.5 Lagging and Leading Indicators

Lagging Indicators can be thought of as 'traditional' safety metrics on the outcome of past accidents that have resulted from particular courses of action. This of course is not ideal when we want to prevent an accident from occurring in the first place. Any accident resulting from the deployment of a RL System would adversely affect the acceptance of such systems within society.

In order for predictive hazard analysis to be undertaken during run-time, monitoring and feedback must be designed into the system such that the system can assess whether it is moving towards a hazardous condition and that a reconfiguration of the system should be enacted in order to mitigate failure.

Thus, in order to effectively identify the potential for system failure, run-time system monitoring needs to be introduced in order to detect 'warning signs' that confidence in the enforcement of safety-related constraints of the system is eroding and that the system is moving towards an unsafe state. Whilst historically these 'warning signs' have, typically, only been apparent after failure has occurred, as stated in [21] the aim of system monitoring is to put in place a means of detecting the migration towards *"a state of unacceptable risk before an accident occurs."*

Accordingly, Leading Indicators need to be identified such that potential failures can be predicted and mitigating actions can be implemented in order to re-establish confidence in the system. This concept supports the claims being made in nodes G3 and G4 of Fig. 2. However, defining Leading Indicators depends on the ability to understand, explain and appropriately constrain the behaviour of the RL System, which is a challenge we explore in the next sections.

4.6 Safe Constraints on Learning

Despite the increasing promise of capability that RL Systems might bring, such systems will still need to comply with strict safety constraints in order to be accepted and deployed. How do we effectively constrain a system that has a seemingly boundless mechanisms to learn, reconfigure and adapt its behaviour?

If learning is limited in prescribed ways, some analysis might be possible, depending on the precise details of the learning limitations. If these are analysed prior to deployment to rule out certain behaviours, perhaps an analysis could show that it would not behave 'dangerously' [22].

However, if we constrain the system's ability to learn then we might also be constraining its capacity to learn what is actually the safest course of action [23]. If we don't allow the system to do something 'wrong' then it may not learn as effectively [20], in the same way that a child learns not to touch a hot surface if it has already endured some pain from a previous experience. If we constrain the system's ability to learn we may be preventing it from learning what 'safe' actually means and restrict the system to our own understanding of 'safe', which in turn might be over-pessimistic.

What we should perhaps consider is exploring the concept of providing a 'safe learning environment' for the RL System in which it can learn, where models of other systems and the interactions with the environment are simulated so that no harm can be caused to humans, assets, or the environment. This approach would support nodes G2 and G3 in Fig. 2. However, this is often complicated by issues around the gap between the simulated and actual environments, including issues related to different societal/human values.

4.7 Fail Safe

To 'fail safe', in the context of a safety-critical system, usually means that the system has determined that it is no longer able to safely provide its intended functionality and that the only safe course of action is to move to a pre-defined 'fail-safe' state, with limited functionality, thus mitigating or preventing any harm to humans, assets or the environment.

This of course is predicated on the RL System's own ability to recognise that it can no longer continue to operate in a 'safe' manner, which would require knowledge and/or definition of 'safe' (node C1 of Fig. 2) along with feedback from the environment (C3), potentially informed by Leading Indicators, and knowledge of what it needs to do in order to transition to a 'Fail Safe' state (nodes G4 and G5). The notions of cost and reward are still likely to be relevant here, e.g. a RL System deciding not to handover control to a human driver until the car speed or environmental conditions are within certain thresholds (i.e. safety benefits of maintaining control outweigh handover safety risks).

4.8 AI-based Safety Assurance and Explainability

Considering we are dealing with systems that have in-built intelligence to adapt their behaviour at run-time in order to meet their functional objective, it perhaps becomes necessary for us to consider applying this same intelligence to the safety assurance side as well. How, as humans, do we believe we are going to be able to rationalise and explain the choices that a machine has made during its course of operation and in meeting its fundamental objective? Considering the DeepMind AlphGo as an example, the developers of the system were at a loss to explain many of the moves that the AlphaGo engine was making in its run-up to defeating the champion Lee Sedol. AlphaGo's learning had moved beyond the limits of what the developers were able to comprehend, which brings about an interesting question when considering the safety assurance of RL Systems; do we need to consider implementing 'Intelligent Safety' whereby the safety argument and monitoring evolves alongside the system rather than the conventional use of run-time monitoring based on predetermined Leading Indicators, e.g. a 'Safety Supervisor' [24]?

The implementation of such a dynamic approach would affect the entire argument framework in the proposed argument structure in Fig. 2 and would offer an idealised, though highly contentious, realisation of the notion of Dynamic Safety Cases [8], i.e. the RL System producing its own safety argument and evidence.

5 Conclusions

ML, and more specifically RL, is a very powerful technique whose adoption within a number of industries is increasingly growing. However, the ability to justify the deployment of such technology into applications of a safety-critical nature poses fundamental challenges. In this paper we have covered what it might mean for a RL System to be considered 'safe' and covered both the 'traditional' (Safety-I) and new (Safety-II) approaches to assuring safety.

In addition, we have presented a high-level safety argument that could be used as the basis of forming a safety case for RL Systems. We have also highlighted a number of key challenge areas that we believe need to be addressed in support of a safety case.

We do however recognise the limitations of the proposed argument structure and the discussion around the challenges. As such we have identified a number of considerations for further work when considering the safety assurance of RL Systems. These are:

1. To what extent should RL algorithms be constrained in the way in which they learn?

2. How would we implement 'Intelligent/Dynamic Safety', whereby the safety monitoring evolves alongside the system?
3. To what extent should a RL System be allowed to create/update its own safety case as it learns?

We hope that progress in answering these questions, in the context of the challenges and argument presented, would be beneficial to the safe adoption of RL Systems into critical applications.

References

1. Faria, J.M.: Non-determinism and failure modes in machine learning. In: 2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), pp. 310–316. IEEE (2017)
2. Calinescu, R.: Emerging techniques for the engineering of self-adaptive high-integrity software. In: Cámara, J., de Lemos, R., Ghezzi, C., Lopes, A. (eds.) Assurances for Self-Adaptive Systems. LNCS, vol. 7740, pp. 297–310. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36249-1_11
3. McDermid, J.: Safety of autonomy: challenges and strategies. In: International Conference on Computer Safety, Reliability, and Security. Springer (2017)
4. McDermid, J.: Playing catch-up: The fate of safety engineering. In: Developments in System Safety Engineering, Proceedings of the Twenty-fifth Safety-Critical Systems Symposium, Bristol, UK (2017). ISBN 978–1540796288
5. Pumfrey, D.J.: The Principled Design of Computer System Safety Analyses. Ph.D. thesis, University of York (1999)
6. Hollnagel, E.: Safety-I and Safety-II: The Past and Future of Safety Management. Ashgate Publishing, Ltd. (2014)
7. Hollnagel, E., Leonhardt, J., Licu, T., Shorrock, S.: From Safety-i to Safety-ii: A white paper. European Organisation for the Safety of Air Navigation (EUROCONTROL), Brussels (2013)
8. Denney, E., Pai, G., Habli, I.: Dynamic safety cases for through-life safety assurance. In: International Conference on Software Engineering (ICSE 2015) (2015)
9. Assurance Case Working Group [ACWG]: GSN community standard version 2. Safety Critical Systems Club (2018)
10. Kelly, T.P.: Arguing Safety - A Systematic Approach to Managing Safety Cases. Ph.D. thesis, The University of York (1998)
11. Porter, Z., Habli, I., Monkhouse, H., Bragg, J.: The moral responsibility gap and the increasing autonomy of systems. In: First International Workshop on Artificial Intelligence Safety Engineering (WAISE) (2018)
12. Suleyman, M.: In 2018, AI will gain a moral compass, January 2018. http://www.wired.co.uk/article/mustafa-suleyman-deepmind-ai-morals-ethics. Accessed 09 Mar 2018
13. Russell, S.: 3 principles for creating safer AI, April 2017. https://www.ted.com/talks/stuart_russell_how_ai_might_make_us_better_people. Accessed 09 Mar 2018
14. Bostrom, N., Yudkowsky, E.: The ethics of artificial intelligence. Camb. Handb. Artif. Intell. **316**, 334 (2014)
15. Dennis, L., Fisher, M., Slavkovik, M., Webster, M.: Formal verification of ethical choices in autonomous systems. Rob. Auton. Syst. **77**, 1–14 (2016)

16. Yampolskiy, R.V.: Artificial intelligence safety engineering: why machine ethics is a wrong approach. In: Müller, V. (ed.) Philosophy and Theory of Artificial Intelligence. SAPERE, vol. 5, pp. 389–396. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-31674-6_29
17. Leong, C., Kelly, T., Alexander, R.: Incorporating epistemic uncertainty into the safety assurance of socio-technical systems. arXiv preprint arXiv:1710.03394 (2017)
18. Rushby, J.: Logic and epistemology in safety cases. In: Bitsch, F., Guiochet, J., Kaâniche, M. (eds.) SAFECOMP 2013. LNCS, vol. 8153, pp. 1–7. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40793-2_1
19. Morris, A.H.: Decision support and safety of clinical environments. BMJ Qual. Saf. **11**(1), 69–75 (2002)
20. Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., Mané, D.: Concrete problems in AI safety. arXiv preprint arXiv:1606.06565 (2016)
21. Leveson, N.: A systems approach to risk management through leading safety indicators. Reliab. Eng. Syst. Saf. **136**, 17–34 (2015)
22. Garcıa, J., Fernández, F.: A comprehensive survey on safe reinforcement learning. J. Mach. Learn. Res. **16**(1), 1437–1480 (2015)
23. Mason, G.R., Calinescu, R.C., Kudenko, D., Banks, A.: Assured reinforcement learning with formally verified abstract policies. In: 9th International Conference on Agents and Artificial Intelligence (ICAART), York (2017)
24. Feth, P., Schneider, D., Adler, R.: A conceptual safety supervisor definition and evaluation framework for autonomous systems. In: Tonetta, S., Schoitsch, E., Bitsch, F. (eds.) SAFECOMP 2017. LNCS, vol. 10488, pp. 135–148. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66266-4_9