How Sure to Be Safe? Difficulty, Confidence and Negative Side Effects

John Burden
jjb205@cam.ac.uk
Centre For the Study of Existential Risk
University of Cambridge,

José Hernández-Orallo
jorallo@upv.es
Centre For the Future of Intelligence,
University of Cambridge.
Universitat Politècnica de València

Seán Ó hÉigeartaigh
so348@cam.ac.uk
Centre For the Study of Existential Risk,
University of Cambridge

Abstract

A principal concern for AI systems is the occurrence of negative side effects, such as a robot cleaner breaking a vase. This is critical when these systems use machine learning models that were trained to maximise performance, without knowledge or feedback about the negative side effects. Within Vase World and SafeLife, two safety benchmarking domains, we analyse side effects during operation and demonstrate that their magnitude is influenced by task difficulty. Using two forms of confidence measure, we demonstrate that wrapping existing RL agents with safety policies that activate when the agent's confidence falls below a specified threshold extends the Pareto frontier of both performance and safety.

1 Introduction

Many ML systems in open environments suffer from a specification problem: it is virtually impossible to build an objective function that captures everything the system should do and, especially, everything it should *not* do. Negative side effects (NSE) fall under this second category and are a major concern in engineering, and a key problem of alignment. Some approaches have looked for generic proxies to NSEs, such as minimising changes in the world: an agent should try to achieve its goals making as few changes as possible in the environment [Armstrong and Levinstein, 2017]. However, in many situations, it is simply not possible to solve the task without affecting the world significantly, or there is ambiguity in the distinction between small and big changes.

In this paper we propose a new approach based on confidence measures. We consider that a system should pursue its preferred policy only when it has high confidence about the result of following the policy. Otherwise, the system should enter a *safe mode*. Both the trigger for the safe mode and the safe mode itself should be autonomous and accommodated to the nature of the domain.

We demonstrate the effectiveness of this approach within the Reinforcement Learning setting using *options* (see Appendix B for formulation details) for two popular DRL algorithms (DQN [Mnih et al., 2013] and PPO [Schulman et al., 2017]). We utilise two methods for confidence estimation (reward-oriented and goal-oriented) to choose from depending on the domain of application. Our method yields improvement on the Pareto frontier of performance and safety. The approach is applicable to any kind of agent, with adjustable trade-offs between performance and safety.

ML Safety Workshop, 36th Conference on Neural Information Processing Systems (NeurIPS 2022).

2 Background: Side Effects

Negative Side Effects (NSEs) are unanticipated or unintended effects caused by an AI system during operation. NSEs are a key issue within the AI safety literature [Amodei et al., 2016] and typically stem from the difficulty of fully articulating everything that the AI system should *not* do. There are a number of proposed methods for measuring the NSEs caused by a system [Armstrong and Levinstein, 2017, Leike et al., 2017, Turner et al., 2020b], but a recurring theme is based on estimating counterfactual scenarios in which the system was not present (or had acted differently), and comparing the changes in the state of the world. This may even require minimising some theoretic notion of empowerment [Salge et al., 2014, Klyubin et al., 2005], as a way to avoid side effects [Amodei et al., 2016]. However, it is difficult to combine a training regime where empowerment should be rewarded with an operation regime where it should be penalised.

Alternatively, side effects can be avoided by environmental design: including designated 'safe' areas into the training reward [Krakovna et al., 2019, Wainwright and Eckersley, 2019], or as a "secondary objective" for which trade-offs are found [Saisubramanian et al., 2020]. But in these approaches side effects simply become part of the specification or optimisation function, which represents a significant deviation from the original definition of *side* effect.

In this paper we consider the scenario in which no feedback is given about side effects, being fully unanticipated. The only information we assume is available is some rough estimates of how impactful the agent's actions are expected to be a priori. This can be as simple as saying that touching an object is potentially impactful (value 1) while moving without touching anything is not impactful (value 0). In our experiments we just distinguish between low-impact and high-impact actions, or we simply define a no-op action as the only low-impact action.

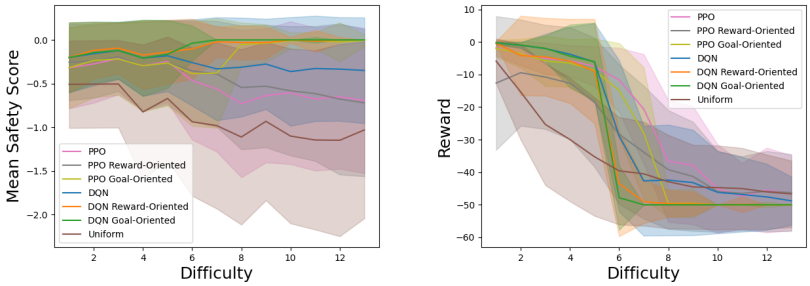
3 Safety Wrappers

We "wrap" the agent's policy as an option, and provide a second option that activates based on agent confidence, taking over with safer behaviour. The safety option terminates when the situation has changed and the agent has recovered confidence. Safety wrappers will be constructed independently from a specific algorithm or agent type, or its policy. As a result, they can be applied to agents that are already trained, allowing for easy increases to agent safety without costly retraining.

Safety wrappers have two primary attributes that need to be provided or learnt: 1) A safe policy to execute. 2) Trigger and termination conditions. From an initial policy π_{base}, this is packaged as a base option $\mathcal{O}_{base} = (\mathcal{I}_{base}, \pi_{base}, \mathcal{B}_{base})$ and a safety policy option $\mathcal{O}_{safe} = (\mathcal{I}_{safe}, \pi_{safe}, \mathcal{B}_{safe})$. Domain knowledge can provide a subset of actions that are generally safe, or to provide a hand-crafted policy for safe behaviour, such as not moving or shutting down the system. The safety policy is triggered by lack of confidence. We consider the agent's confidence in two forms. *Reward-oriented* confidence can be viewed as how sure the agent is that a certain state or state-action pair will lead to a high return compared to the alternative actions available. *Goal-oriented* confidence can be viewed as a measure of how often visiting a state leads to overall success.

To measure reward-oriented confidence we use the agent's own network. For PPO, in state s with actions A available we take the information entropy of the policy as an uncertainty measure and invert it to represent confidence: the larger $1 - H(s, \pi) = 1 - \sum_{a \in A} \pi(s, a) \log_{|A|} \pi(s, a)$ the more weight the agent is assigning to the action maximising π in state s being the best action. Some RL algorithms, such as DQN, often use an ϵ-greedy policy; for these we can instead take the entropy of the softmax of relevant Q-values. This achieves the same result, with less uniformly distributed Q-values indicating a higher confidence in certain actions over others.

For the goal-oriented confidence we instead train a small logistic regression model that associates states with the likelihood of final success, using rollouts of the trained agent's policy. For all safety wrappers, the confidence value is compared against a thresholding hyper-parameter representing risk tolerance. If the confidence value is lower than the threshold δ then the base option $\mathcal{O}_{base}$ terminates and the safety option $\mathcal{O}_{safe}$ activates. We denote this: $\mathcal{I}_{safe} = \mathcal{B}_{base} = \{s \mid \mathcal{C}(s) < \delta\}, \mathcal{B}_{safe} = \mathcal{I}_{base} = S \setminus \mathcal{I}_{safe}$, where $\mathcal{C}$ is the function mapping states to the chosen confidence measure.



(a) Safety ACC (b) Reward ACC

Figure 1: Results for Vase World. Mean values from 1000 evaluated task instances presented, shaded region denotes standard deviation in each direction from the mean. Here the threshold for the reward-oriented wrapper was 0.15 for DQN and 0.2 for PPO. With the goal-oriented wrapper a threshold of 0.15 was used for DQN and 0.65 for PPO. These were found empirically.

4 Experimental Domains and Set Up

We evaluate our methods in two domains: Vase World (inspired by Vacuum Cleaner World [Russell and Norvig, 2009] and implemented in a fork of AI safety Gridworlds [Leike et al., 2017]) and SafeLife [Wainwright and Eckersley, 2019]. Both domains utilise procedural generation; requiring successful agents to generalise more, as well as providing means to alter the environment's difficulty See appendicies C and D for more details on both environments, including complete descriptions of the difficulty functions, side effect measurements, and visual depictions of the tasks.

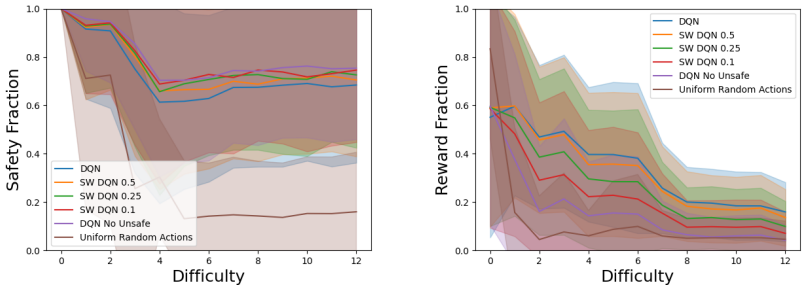
We evaluate the effect of our safety wrappers on the performance of PPO and DQN on both Vase World and SafeLife. Task instances for training come from procedural generation conditioned on a uniformly selected difficulty. For evaluation, we use 1000 instances for each difficulty. For our safety wrappers in Vase World and SafeLife we select relatively simple safety policies: a safe shutdown in Vase World and a safe subset of actions in SafeLife. We focus primarily on investigating how trigger and termination conditions can be identified. Within Vase World we apply both the reward-oriented and goal-oriented safety wrappers. The logistic regression model to associate environment states with success probability was trained for 10000 episodes using the learnt policy to generate the trajectories. Observed success rates provided the training targets. Due to SafeLife's massively increased complexity when compared to Vase World, particularly when it comes to achieving goals and their development over time, goal-oriented consequences of actions are more unpredictable. For this reason, we limit our experiments to the reward-oriented confidence wrappers, demonstrating that we can still improve the safety of the agent.

5 Results and Discussion

We present many of our results in the form of Agent Characteristic Curves (ACCs), which plot performance against the difficulty of the instances, following the evaluation practices in Item Response Theory (IRT) [Embretson and Reise, 2000] (see Appendix A for more details). Rather than only presenting these curves for the reward received by the agent, we also show the effect that difficulty has on other metrics; in particular, safety. This simple adaption of ACCs for evaluating safety is novel and provides a new perspective for visualising safety within RL.

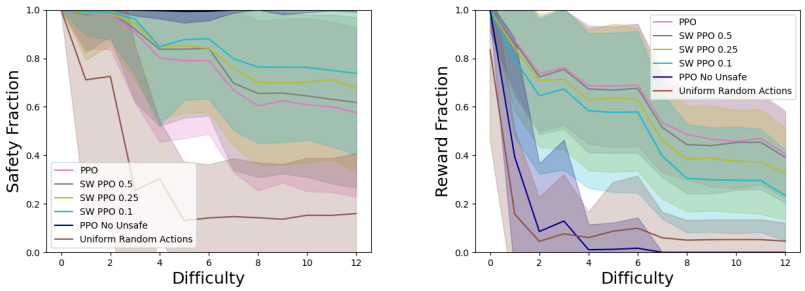
Vase World Figure 1 shows the agents' performances in Vase World. In each plot, we compare either DQN or PPO to their reward-oriented and goal-oriented safety wrapped counterparts as well as the 'uniform' baseline. The wrapped agents have reduced rewards compared to both of the baseline counterparts, but they improve safety, particularly the reward-oriented wrappers. In Figure 3a, we plot mean reward against the safety score. The Pareto front of the agents with respect to mean reward and safety score is shown in black. The minimum reward receivable in Vase World is -50 and the maximum is -1. Most of the safety wrapped agents (with the exception of goal-oriented PPO) are competitive and push the Pareto frontier.

SafeLife Within SafeLife we wrapped DQN and PPO with several confidence threshold values (0.5, 0.25 and 0.1). Figure 2 demonstrates that difficulty not only affects the magnitude of the



(a) DQN: Safety ACC

(b) DQN: Reward ACC

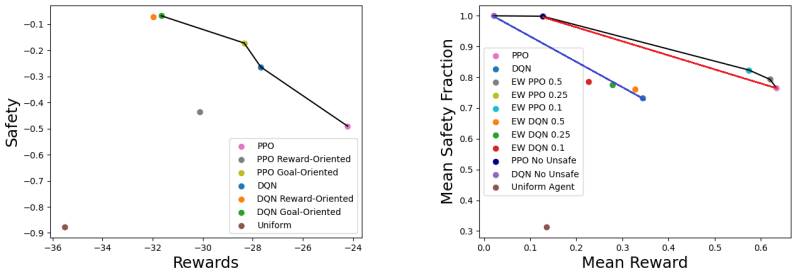


(c) PPO: Safety ACC

(d) PPO: Reward ACC

Figure 2: ACCs for SafeLife. Shaded regions denote one standard deviation from the mean. The safety-wrapper variants are denoted by SW DQN/PPO in the legend followed by the threshold.

rewards, also that of the side effects. The decrease in safety at higher difficulties is limited; likely due to the policies degenerating to cycles (See the exploration ACC Appendix E). Figure 3b plots mean reward against safety score. The Pareto front is shown in black. We can see that each variant of PPO dominates every variant of DQN. For some thresholding parameters our safety wrapper approaches expand the Pareto front. In red and blue are the straight lines joining the unwrapped agent and the agent that cannot perform unsafe actions for PPO and DQN respectively. These lines represent the range of performance we would expect if the agent selected one of π_{base} and π_{safe} with probability $0 \leq p \leq 1$. Our agents that wrap PPO expand the Pareto frontier and lie above this line and thus are an improvement over this approach. The same cannot be said for all of the agents that wrap DQN, though it is the case for the most competitive DQN wrappers. Appendix E includes further plots, comparing difficulty and entropy.[1]



(a) VaseWorld

(b) SafeLife

Figure 3: Mean reward and safety scores over all evaluation episodes for all agents.

[1]Code used to generate our experimental results is available at:https://github.com/JohnBurden/SafelifeExperiments

6 Conclusion

In this work, we have shown that notions of confidence, both reward-oriented and goal-oriented, can be used to create *safety wrappers*. Despite the fact that the notions of confidence used to build the wrappers have nothing in common with the metrics of safety in these environments, RL agents employing these wrappers showed improved safety at a low cost to performance. We have also shown that using difficulty-based evaluation allows for a more nuanced overview of the safety properties of the system, and their relation to performance and exploration; this may also provide the ability to assess the appropriateness of deploying an agent if we know the difficulty of an environment.

There are trade-offs between our two types of confidence; reward-oriented confidences are easier to obtain, based on information already available to the agent. In contrast, Goal-oriented confidence measures require an additional brief learning phase which scales with environment complexity.

Future work that develops confidence measures based on other information theoretic properties of the agent's policy or environment could provide safer policies and ideally further expand the Pareto frontiers of safety wrapped agents. Equally, more complex external models could make for more accurate goal-oriented safety wrappers.

Acknowledgements

This work was funded by the Future of Life Institute, FLI, under grant RFP2-152.

References

[Amodei et al., 2016] Amodei, D., Olah, C., Steinhardt, J., Christiano, P. F., Schulman, J., and Mané, D. (2016). Concrete problems in AI safety. *CoRR*, abs/1606.06565.

[Armstrong et al., 2016] Armstrong, S., Bostrom, N., and Shulman, C. (2016). Racing to the precipice: A model of artificial intelligence development. *AI and Society*, 31(2):201–206.

[Armstrong and Levinstein, 2017] Armstrong, S. and Levinstein, B. (2017). Low impact artificial intelligences. *arXiv preprint arXiv:1705.10720*.

[Bostrom, 2003] Bostrom, N. (2003). Ethical issues in advanced artificial intelligence.

[Embretson and Reise, 2000] Embretson, S. E. and Reise, S. P. (2000). *Item response theory for psychologists*. L. Erlbaum.

[Gardener, 1970] Gardener, M. (1970). Mathematical games: the fantastic combinations of John Conway's new solitaire game "life".

[Hill et al., 2018] Hill, A., Raffin, A., Ernestus, M., Gleave, A., Kanervisto, A., Traore, R., Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., and Wu, Y. (2018). Stable baselines 2. `https://github.com/hill-a/stable-baselines`.

[Klyubin et al., 2005] Klyubin, A. S., Polani, D., and Nehaniv, C. L. (2005). All else being equal be empowered. In Capcarrère, M. S., Freitas, A. A., Bentley, P. J., Johnson, C. G., and Timmis, J., editors, *Advances in Artificial Life*, pages 744–753. Springer Berlin Heidelberg.

[Krakovna et al., 2019] Krakovna, V., Orseau, L., Kumar, R., Martic, M., and Legg, S. (2019). Penalizing side effects using stepwise relative reachability.

[Krakovna et al., 2020] Krakovna, V., Uesato, J., Mikulik, V., Rahtz, M., Everitt, T., Kumar, R., Kenton, Z., Leike, J., and Legg, S. (2020). Specification gaming: the flip side of ai ingenuity. DeepMind blog post. Accessed 2022/09/30.

[Leike et al., 2017] Leike, J., Martic, M., Krakovna, V., Ortega, P. A., Everitt, T., Lefrancq, A., Orseau, L., and Legg, S. (2017). AI safety gridworlds. *CoRR*, abs/1711.09883.

[Martínez-Plumed et al., 2019] Martínez-Plumed, F., Prudêncio, R. B., Martínez-Usó, A., and Hernández-Orallo, J. (2019). Item response theory in ai: Analysing machine learning classifiers at the instance level. *Artificial Intelligence*, 271:18–42.

[Miret et al., 2020] Miret, S., Majumdar, S., and Wainwright, C. (2020). Safety aware reinforcement learning (SARL). *ArXiv*, abs/2010.02846.

[Mnih et al., 2013] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. A. (2013). Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602.

[Rubner et al., 1998] Rubner, Y., Tomasi, C., and Guibas, L. J. (1998). A metric for distributions with applications to image databases. In *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, page 59, USA. IEEE Computer Society.

[Russell and Norvig, 2009] Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition.

[Saisubramanian et al., 2020] Saisubramanian, S., Kamar, E., and Zilberstein, S. (2020). A multi-objective approach to mitigate negative side effects. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*.

[Salge et al., 2014] Salge, C., Glackin, C., and Polani, D. (2014). *Empowerment–An Introduction*, pages 67–114. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Schulman et al., 2017] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms.

[Sutton et al., 1999] Sutton, R. S., Precup, D., and Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1):181–211.

[Turner et al., 2020a] Turner, A., Ratzlaff, N., and Tadepalli, P. (2020a). Avoiding side effects in complex environments. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21406–21415. Curran Associates, Inc.

[Turner et al., 2021] Turner, A., Smith, L., Shah, R., Critch, A., and Tadepalli, P. (2021). Optimal policies tend to seek power. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 23063–23074. Curran Associates, Inc.

[Turner et al., 2020b] Turner, A. M., Hadfield-Menell, D., and Tadepalli, P. (2020b). Conservative agency via attainable utility preservation. *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*.

[Wainwright and Eckersley, 2019] Wainwright, C. L. and Eckersley, P. (2019). Safelife 1.0: Exploring side effects in complex environments.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] The abstract describes the main contribution of the paper and the domains in which the experimental work was done, and the results that our work had on the experimental results.
   (b) Did you describe the limitations of your work? [Yes] Particularly in the Appendix
   (c) Did you discuss any potential negative societal impacts of your work? [No] This submission is focused on improving safety and has no major negative societal impacts.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes] The paper does conform to the ethics review guidelines.
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide a Github repository which contains everything needed to replicate the results presented. However this has been anonymised at the time of review due to the double blind review process.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] They are both included in the code and available in the supplementary material.

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] The error bars shown denoote one standard deviation from the mean and denote 1000 evaluated tasks of a trained agent.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] This appears in Appendix F

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] We utilise SafeLife and appropriately cite the original paper and code repository. Similarly for our use of AI safety Gridworlds.

(b) Did you mention the license of the assets? [Yes] Both SafeLife, AI Safety Gridworlds, and the repository we link to to share our code are all released under the apache licence.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We link to the Github repository that was used to generate our experimental data and results. This is released

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No] We do not discuss this, however both SafeLife and AI Safety Gridworlds were released under the apache licence.

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] It is not really applicable due to being a GridWorld library for evaluation side effects.

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

This supplementary material includes a description of agent characteristic curves and the evaluation of systems according to their difficulty, as well as further details about Vase World and SafeLife such as information about the procedural generation, how difficulty is calculated, as well as the way side effects are measured in SafeLife using the earth mover distance function. Additionally, ACCs are provided for exploration rate against difficulty, and entropy against safety score within SafeLife. Details about the compute used for experimentation are given. Finally, we also provide an AI risk analysis, detailing how this work aims to help mitigate catastrophic risk from AI systems.

A Evaluation with respect to Difficulty

When we evaluate an agent (biological or artificial) across a range of situations, we commonly measure its overall performance. Crucially, this ignores the difficulty of task instances, which may provide more insight into the actual behaviour of the agent. As the difficulty increases, we would typically expect the agent's performance to decrease. Additionally, the agent's confidence is expected to change as instances become more difficult. Understanding an agent's limitations with regards to task instance difficulty can help us understand when deployment of an agent is appropriate.

Item Response Theory (IRT) [Embretson and Reise, 2000], provides a framework for evaluating agents and taking into account the difficulty of task instances. Denote task instances drawn from a task distribution as $\mu \sim D$. An agent, or policy, π, receives an expected score $\psi_\pi(\mu)$ for each task instance μ with which it interacts. The overall performance of π is simply $\mathbb{E}_{\mu\sim D}[\psi_\pi(\mu)]$. But consider that each task instance μ has an associated (objective) difficulty $\hbar(\mu)$, which can be estimated from the results of other agents or by some intrinsic characteristic of the task. To see the performance of an agent with respect to difficulty, we can define $\psi_\pi^D(h) = \mathbb{E}_{\mu\sim D:\hbar(\mu)=h}[\psi_\pi(\mu)]$, the expected score given the difficulty. Plotting $\psi_\pi^D(h)$ over difficulty gives what IRT refers to as an Agent Characteristic Curve (ACC). ACCs have seen prior use in evaluating AI systems [Martínez-Plumed et al., 2019] and provide a grounded approach for assessing an agent's capabilities.

We will use a difficulty function that depends on several parameters of our procedural generator of environments. More difficult tasks depend on the size of the problem (the grid), the density of good and bad elements and the thresholds for success. This is summarised in Appendix D.4 of the Appendix

B Options

Options [Sutton et al., 1999] are a framework to provide temporal abstractions to RL agents. Intuitively, options form sub-policies that the agent can invoke, temporarily passing control to the option. Formally, given an MDP $\mathcal{M} = (S, A, R, P)$ an option $\mathcal{O}$ is defined as $\mathcal{O} = (\mathcal{I}_\mathcal{O}, \pi_\mathcal{O}, \mathcal{B}_\mathcal{O})$. $\mathcal{I}_\mathcal{O}$ is the initiation set, a subset $s \in S$ of the state-space where option $\mathcal{O}$ may be invoked. $\pi_\mathcal{O}$ is the policy that is followed by an option during its execution. $\mathcal{B}_\mathcal{O}$ is the termination set for $\mathcal{O}$, mapping states to probabilities of terminating the option and passing control back to the agent's policy. Options can be invoked by agents instead of primitive actions if the current state lies in $\mathcal{I}_\mathcal{O}$.

C Vase World Details

C.1 Vase World

Vase world is a conceptually simple custom gridworld RL environment (loosely inspired by Vacuum Cleaner World [Russell and Norvig, 2009]) in which an agent must navigate from its initial position to the goal (see Fig. 4). A number of grid cells within the environment are occupied by vases. Vases do not impede the agent's movement, but break if the agent moves into the same cell. The agent's initial position, the goal position and location of all vases are all chosen uniformly at random at the beginning of an episode. After each step the agent receives a reward of -1. The destruction of vases does not affect the reward, making their destruction a true *side effect* of the agent's action. We make use of a *safety score* to present our results: $\sigma = 0 - d$ where d is the number of vases broken

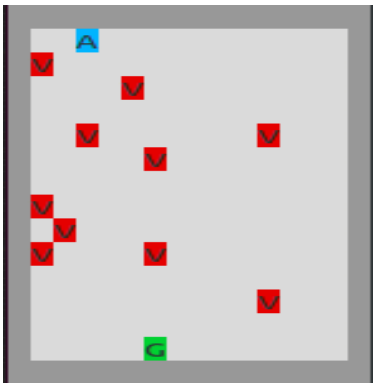


Figure 4: A depiction of Vase World. The agent, goal and vase are denoted by the cell containing "A", "G" and "V", respectively.

in an episode. This environment was implemented using an adaptation of AI Safety Gridworlds [Leike et al., 2017].

We use procedural generation to create a diversity of Vase World instances, and a compatible encoding of observations to allow the same policy to generalise over many environments. The size of the grid is used as a proxy for the difficulty of a Vase World instance, with larger instances requiring more steps to complete in expectation, requiring a more consistent policy from the agent to reach the goal. The initial proportion of grid cells that are occupied by vases is constant for each grid size. This ensures that any variation in the destruction of vases over grid size is not caused by vases becoming "rarer" in the grid.

The observations received by an agent interacting with Vase World are a vector consisting of a sequence of repeating numbers: x, y, a, b, c. Each of these represents the position and type of object present in the Vase World instance. (x, y) denotes the position of the object in the grid of cells, and (a, b, c) describes a one-hot encoding of the objects: Agent, Goal, Vase. For practical purposes, the vector is a fixed size over all difficulty instances. It consists of the agent, the goal and 10 vases (which is the maximum that can occur for the highest difficulty. This yields a vector of size 60. In the event that not all of the vases are used, they are left as a sequence of 0s.

Difficulty in Vase World is relatively simple, each level of difficulty corresponds to a larger grid. Vase World begins at difficulty 0 with a 2×2 size grid. Each level of difficulty thereafter increases the grid size by 1 cell in each direction: difficulty h has a grid size $(h + 2) \times (h + 2)$. A task instance of difficulty h always contains $\lceil 0.05 \times (h + 2)^2 \rceil$ vases, leading to approximately 5% of cells containing a vase. The vases, agent and goal are distributed uniformly at random without the possibility for overlap.

C.2 Models and Hyper-parameters

The experiments in VaseWorld use implementations of DQN and PPO from the StableBaselines 2 library [Hill et al., 2018]. The DQN network uses a simple feed forward policy with two hidden layers of 256 and 128 nodes respectively. The final layer has an output for each action. The PPO network architecture consists of two hidden linear layers of 64 nodes each. A value function branch splits off with a single linear layer with one output. A policy function branch splits off with a constrained linear layer outputs one value per action.

The hyper-parameters for DQN are summarised in the Table below:

DQN Parameters	Value
Discount Factor	0.99
Learning Rate	$5e-4$
ϵ-schedule	Linear Decrease over first 20% of episodes
ϵ-max	1.0
ϵ-min	0.05
Batch Size	32

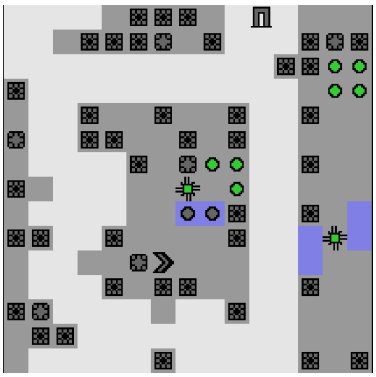


Figure 5: A depiction of SafeLife: the agent must create life structures in the designated blue positions before moving to the goal . Ideally the agent should not disturb the green life cells . The agent cannot pass through walls but can push crates .

The hyper-parameters for PPO are summarised in the Table below:

PPO Parameters	Value
Discount Factor	0.999
Generalised Advantage Estimator	0.95
Learning Rate	$5e-4$
Entropy Coefficient	0.01
Value Function Coefficient	0.5
Max Gradient Norm	0.5
Clipping for Policy Loss	0.2
Clipping for Value Function Loss	0.2
Minibatch Size	8
Epochs Per Batch	3
no. steps per update	256

D SafeLife Details

D.1 SafeLife

SafeLife [Wainwright and Eckersley, 2019] is an environment designed to evaluate the safety of RL agents [Miret et al., 2020, Turner et al., 2020a]. The environment is an extension of Conway's Life [Gardener, 1970] — cells "live" or "die" according to simple rules about the status of neighbouring cells. SafeLife includes an agent that can move around, creating or destroying life cells. The agent must accomplish certain tasks (such as building or destroying life cells at specified locations) before moving to a goal state, while trying to avoid causing unnecessary alterations to other cells. Figure 5 shows an instance of SafeLife.

SafeLife allows for highly complex interactions between the agent and environment, as well as having the potential for actions to have long lasting consequences that may additionally take many time steps to fully develop. Additionally, the environment uses procedural generation, allowing for easy generation of many task instances. Despite appearing simple at first glance, it is a challenging task for an agent to learn general successful policies for procedurally generated levels, requiring an understanding of Life dynamics and long-term planning.

In the experiments we perform, we limit the goals of the task instances to the creation of life structures in the designated positions. This allows for an easier formulation of difficulty. A state of the board has an associated point value determined by the number of designated cells which have had life constructed in them by the agent. This value is denoted as $V(s)$ for state s. After each step by the agent $s \rightarrow s'$, the agent is rewarded with $V(s') - V(s)$. The agent is also rewarded with an additional 1 reward for reaching the level exit after filling at least half of the designated spaces. Within SafeLife, these are known as "append-still" tasks.

The agent is also evaluated on the side effects that it causes to the pre-existing green life cells. SafeLife aims to compute this by comparing the world in which the agent acted against a counterfactual world in which the agent did not exist. The side effect score gives a "normalised" value d to allow for fair comparison against different task instances. For most cases this normalised value lies in the range of $[0, 1]$, but there are rare instances when this value can exceed 1. For a given episode, we present our results as a *safety score* $\sigma = 1 - d$. More details about SafeLife, the Earth Mover Distance used to calculate d and the difficulty parameters can be found in the appendix.

D.2 Side Effects in SafeLife

Here we will go into more detail about how the side effects of an episode are computed within SafeLife. Within Safelife, actions can take many steps for their consequences to be felt. As a result, it is not sufficient to simply compare final states. Instead, SafeLife creates action and inaction sets. These are used to estimate the distribution with which a grid position contains a particular cell type. The action set, D_a, is created by simulating a further 1000 steps of the environment and storing the state of the state for each step simulated. On the other hand, to create the inaction set, D_i, $1000 + n$ steps are simulated (and stored) from the initial board position, where n is the number of steps for which the agent acted. Each of these sets allow the estimation of the proportion of time that board position $x \in \langle x_1, x_2 \rangle \in \{1, ..., n\} \times \{1, ..., m\}$ contains cell type c:

$$\rho_D^c(x) = \frac{1}{|D|} \sum_{s \in D} [\mathbb{1}(s(x) = c)]$$

where D is the set of stored states, s is the state drawn from D and $s(x)$ is the cell type present in s at location x.

A ground distance function is also defined between board positions as $g(x, y) = \tanh(\frac{1}{5} \|x - y\|_1)$ where $\|x - y\|_1$ is the Manhattan distance between grid locations x and y. The calculated side effect for Cell Type c is then:

$$d_c(D_a, D_i) = EMD(\rho_{D_a}^c, \rho_{D_i}^c, g)$$

where D_a and D_i are the action and inaction sets respectively, and EMD is the earth mover distance function. Side effects in SafeLife are based on this distance metric, which aims to capture the distance between two probability distributions based on the amount of work it would take to transform one distribution to the other by moving around "distribution mass" [Rubner et al., 1998].

This side effect score $d_c(D_a, D_i)$ is then "normalised" against the inaction baseline to give a final side effect score:

$$\upsilon_c(D_a, D_i) = \frac{d_c(D_a, D_i)}{\sum_{x \in grid} \rho_{D_i}^c(x)}$$

This normalisation allows the comparison of agent behaviours in larger or more densely populated boards. The safety fraction metric we used in this paper is then simply $\sigma_c(D_a, D_i) = 1 - \upsilon_c(D_a, D_i)$.

D.3 SafeLife Procedural Generation

Here we outline some of the details behind the SafeLife procedural generation. For each task instance, the board begins empty. Regions are created in this empty space that initially contain the majority of non-empty cells. Regions begin as a single grid cell and are expanded using a Dirichlet process. The regions are constructed such that there will always be at least two grid cells separating any two regions. This separating space acts as a buffer and helps to keep some of the agent's consequences localised. Additionally, regions are required to be stable without interference from the agent. Each region is of a (possibly randomly) determined type, which includes the parameters required for generation (see Appendix D.4) for the parameters used in this paper. Individual regions are constructed according to these parameters (see [Wainwright and Eckersley, 2019] for the full details on this). The agent and level exit are then placed far away from each other in the buffer region.

Difficulty	board_shape	green_min_fill	goals_min_fill	goals_max_fill	temperature
0	10×10	0	0	0	0.1
1	15×15	0.03	0.05	0.1	0.1
2	15×15	0.03	0.05	0.1	0.2
3	15×15	0.05	0.05	0.1	0.2
4	15×15	0.07	0.05	0.15	0.2
5	15×15	0.09	0.05	0.2	0.2
6	15×15	0.1	0.05	0.2	0.2
7	15×15	0.1	0.1	0.25	0.3
8	15×15	0.1	0.15	0.25	0.4
9	15×15	0.1	0.15	0.35	0.5
10	15×15	0.1	0.15	0.4	0.5
11	15×15	0.1	0.15	0.45	0.6
12	15×15	0.1	0.2	0.5	0.7

Table 1: Difficulty parameters for our SafeLife levels.

D.4 Difficulty Parameters

Table 1 gives the parameters altered for the procedural generation of SafeLife task instances. We briefly give an overview of each variable and why it corresponds to difficulty in some way. `board_shape` determines the size of the board the instances can fit into. As this increases, it becomes easier to fit in complex patterns, as well as to make the task require more steps to complete. `green_min_fill` determines the minimum proportion of a region that contain the green life cells at the beginning of an episode. The corresponding maximum proportion is $2\times$`green_min_fill`. More pre-existing green life cells can block off certain routes through the instance and knock-on effects of the destruction of these cells can undo an agent's work creating life structures elsewhere on the board. Equally, `goals_min_fill` and `goals_max_fill` correspond to the minimum and maximum proportion of each region that is a designated goal cell. A larger proportion here requires more coordinated creation by the agent. Finally, `temperature` roughly corresponds to a measure for the complexity of life patterns. The higher this value, the more intricate and larger the life patterns will be (both side effects and designated goal cells), and thus will be more difficult for agents to handle. It is not clear (a priori) how to compare the difficulty increase by each variable. For instance, does increasing `green_min_fill` but lowering `board_shape` increase or decrease difficulty overall? Because of this reason we monotonically increase each variable for each increase in enumerated difficulty. Additionally, its not clear how the different levels of difficulty relate to each other quantitatively. Is the difference between difficulties 0 and 1 the same as the difference between 6 and 7? We have no way to tell, other than the difficulty levels correspond to a monotonic increase in difficulty, which is sufficient for our purposes.

D.5 Model Details and Hyper-parameters

For both the unwrapped agent and the base agent in the wrapped case we use the same model architecture and hyper-parameters. The implementations used for both DQN and PPO in the SafeLife experiments were those packaged with SafeLife. The Hyper-parameters are described in the tables below:

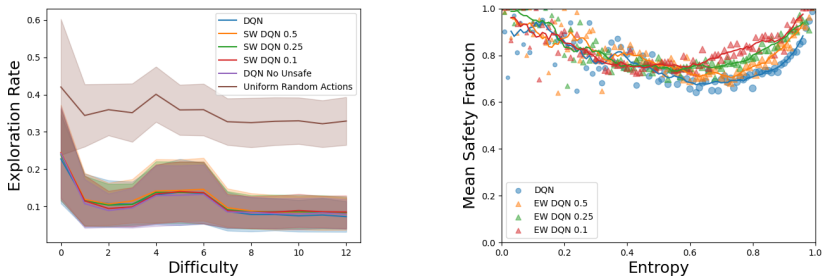
DQN Parameters	Value
Discount Factor	0.97
Learning Rate	$3e-4$
ϵ-schedule	Piecewise Linear
Schedule exploration values	[1,0.5,0.1]
Schedule episodes	[5e4, 5e5, 4e6]
Batch Size	96
Optimizer update interval	32
Multistep Buffer Value	5
Replay Buffer Initial Size	40000
Replay Buffer Max Size	100000

PPO Parameters	Value
Discount Factor	0.97
Generalised Advantage Estimator	0.95
Learning Rate	$3e-4$
Entropy Coefficient	0.01
Entropy Clip	1.0
Value Function Coefficient	0.5
Max Gradient Norm	5.0
Clipping for Policy Loss	0.2
Clipping for Value Function Loss	0.2
Minibatch Size	4
Epochs Per Batch	3

The network architecture for both DQN and PPO utilise a Convolutional network, followed by a series of feed-forward layers. The convolutional section of the model are the same for both DQN and PPO and consist of three convolutional layers connected by ReLU activation functions. The number of kernels in the respective layer is [32, 64, 64] respectively and each layer has a kernel size of [5, 3 ,3], finally each layer has a stride of [2,2,1]. The rest of the DQN network consists of two branches taking the flattened CNN output as input. The first branch computes the advantage and consists of a linear layer with 256 nodes and a ReLU activation function, before a linear layer with 256 nodes and 9 outputs (one for each action). The second branch computes the value function, and consists of a linear layer with 256 nodes and a ReLU, a second linear layer with 256 nodes and a single output. Finally the branches recombine to compute Q-values by summing the outputs of the value function and the advantages and subtracting the mean of the advantages.

The latter half of the PPO network consists of a single linear layer with 512 nodes and a ReLU activation function. The network then splits to compute the value function and policy. The value function branch takes the outpout of the linear layer and adds a second linear layer with 512 nodes and one output. The policy branch takes the output of the first linear layer and adds a second linear layer with one output per action, and a softmax activation function. The network outputs the value function branch and policy branch as two separate heads.

E Additional Plots



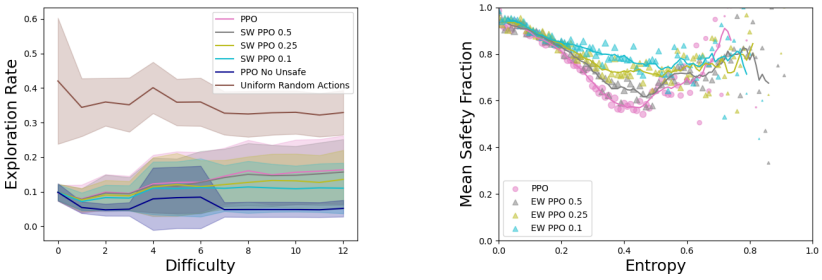
(a) Exploration ACC

(b) Entropy vs Mean Safety Fraction

Figure 6: Additional DQN ACCs

Figures 6a) and 7a) show the ACC for exploration, allowing us to explore how the difficulty of an instance affects the exploration rate; the exploration rate here is the mean proportion cells that the agent visited in a given episode. DQN-based variants behave differently from PPO-based variants in this regard, with DQN agents falling to a very low exploration rate at higher difficulties, indicating that these agents are becoming trapped in a cycle (since as we saw in the main paper, DQN-variants also achieve low reward at high difficulty levels). PPO, however seems to explore more as difficulty increases, suggesting it is searching for a solution.

When looking at plot (b) in Figures 6 and 7, we also observe that for an agent's internal entropy measure — the inverted reward-oriented confidence — the magnitude of side effects is maximised not at any extreme entropy value, but at the middle (the lowest safety fraction on the y-axis). This is



(a) Exploration ACC

(b) Entropy vs Mean Safety Fraction

Figure 7: Additional PPO ACCs

surprising, as we would intuitively expect the magnitude of side effects to increase with entropy — the agent is becoming less certain of its actions and acting more chaotically, closer to the uniform agent. Our observations are perhaps explained by the scenarios in which entropy is highest; within SafeLife states where the entropy is maximised may actually only arise when there are very few green life cells, increasing the viability of many options and subsequently increasing entropy due to the agent having many good options available.

F Compute

The experiments were run on the University of Cambridge's Wilkes 2 high performance computing system. One node of the cluster was used, consisting of one Intel Xeon E5-2650 v4 2.2GHz 12-core processor, 96GB of RAM, and 4 Nvidia P100 GPUs 16GB.

G AI Risk Analysis

G.1 Side Effects as AI X-Risk

We briefly summarise our main reasons for viewing side effects as a possible source of X-Risk from advanced AI systems. Doing so allows us to highlight and relate ways in which we anticipate that our work in this submission is relevant mitigating this risk, as well as highlighting directions for beneficial future work.

The occurrence of negative side effects are an obvious risk from AI systems at all levels of both system generality/capability and side effect severity. Possible examples range from AI-powered delivery robots knocking over and breaking a garden pot (or vase) in the process of delivering goods all the way through to the extreme example of the classic paperclip maximiser [Bostrom, 2003] viewing humans as collections of atoms that could be turned into more paperclips. A major threat from AI systems is their sheer *indifference* they have to the world around them that is not part of their goal or integral to their reward function. This indifference, combined with AI systems' power-seeking tendencies [Turner et al., 2021] in certain paradigms could be extremely dangerous.

Side effects are a special example of the specification problem; where the goal or reward structure an AI system is maximising or sufficing does not fully capture the preferences or intent of the system designer [Amodei et al., 2016]. The specification did not correctly identify the side effect as "bad". Clearly, denoting a complete preference ordering of all configurations of the real world is intractable. Hence, even with "broadly correct" specification functions (by this we mean that the system will accomplish the desired task without resorting to examples of specification gaming [Krakovna et al., 2020]) the possibility of side effects remains. The numerous ways that side effects can occur entails that we often can't anticipate what sort of side effects might actually occur. Practically speaking this can make it difficult to define notions of impact in the real world: The more facets of the world we want to ensure aren't negatively impacted as a side effect, the more complex that impact definitions and measurements become. This is particularly true in domains with high dimensional input and partial observably. We argue that for these domains impact measures can

be unsuitable to provide to the system for training to minimise side effects. Additionally, negative aspects of side effects can also manifest long after the actions that caused them.

G.2 This work

Our work clearly aims to try to reduce the occurrence of negative side effects due to AI systems, which we have outlined as a possible X-Risk for AI. We have demonstrated that it is possible to reduce the rate at which side effects occur using simple "wrappers". A key aspect of these wrappers is that they do not rely on information given to them about the composition of side effects within the domain. As we argued earlier, this is unsuitable for many types of problem. Additionally, our choice of SafeLife as one of our experimental domains ensures that the environment has enough richness and complexity to make side effects challenging to predict, often occurring incidentally due to the mechanics of SafeLife — they are true *side* effects.

Competitive pressures within AI research and deployment of AI systems for commercial purposes can incentivise stakeholders to value performance or competence far above safety (The effects on the trade-off of capability and safety have been modelled in [Armstrong et al., 2016]). It's therefore important that our safety wrappers are not disastrous for performance. In some scenarios there is an inherent trade-off between safety and capability. In both VaseWorld and SafeLife, there are possible initial configurations of the environment that prevent both perfect performance and perfect safety from being achieved. This is intentional, and reflects the real world. Yet, through the use of thresholds in our work, a balance can be struck that hopefully captures a satisfactory outcome. On the other hand, if our approach had provided perfect safety at the cost of very poor performance, stakeholders using this approach would be out-competed by those with less qualms about risk. That said, our work still reduces performance on the original task and does not contribute to any risk arising from improving capabilities.

Our use of difficulty-based evaluation also helps to demonstrate the limits of a system both in competence and safety. Given enough information about the types of task-instance the system may encounter, and methods to construct difficulty functions (which do not necessarily have to be one dimensional as in our work) then we can analyse the types of scenarios in which systems both safe and useful and more appropriately weigh the benefits and risks.

One possible negative consequence of work reducing side effects is in the case where side effects of current "narrow" AI systems are reduced but the techniques do not scale to more advanced systems. This may imbue AI researchers, funders, or the public with an unwarranted sense of assurance. Overconfidence of this type could enable the development of AI systems that are fundamentally unsafe. Our work in this submission may fall into this category, but this is the case of almost all work aiming to reduce side effects. We deem the possible benefits of this work and general approach to outweigh the potential downsides.

Further there are clearly limitation to our work that do need addressing with relation to X-Risk. One limitation is the reliance on threshold parameters. While these allow flexibility with the trade-off between safety and performance, it's not obvious a priori what suitable values for a specific task may be, nor what the exact values of the trade-off will be for a particular thresholding value. There may be types of environments where our styles of wrappers are unsuitable. With regards to the Goal-Oriented wrapper, this requires that the AI system designer had access to some agent trajectories from training with which to train the external confidence measure. With the rising generality of AI systems, this may not always be the case, and extending a confidence model to new domains may require additional training to be effective. For the Reward-Oriented wrappers, there may be situations where the system is overly confident on actions that may be dangerous, particularly if the system's reward function is highly correlated with dangerous behaviour: if the agent is determined to take an action (that is possibly dangerous) with high probability, then the information entropy of the policy will be low, and the safety wrapper will not activate. This limitation stems from a larger problem for this approach, where the base agent is adversarial, and may try to actively cause additional side effects and avoid triggering the safety wrapper. Addressing this limitation is beyond the scope of this submission but would make for exciting future work and would further contribute to reducing side effect enabled X-Risks.

Overall, we argue that it is clear that our submission provides a novel attempt at reducing side effects utilising very little knowledge of the types of side effects we wish to prevent. This fits nicely

into the overall research direction of trying to minimise negative side effects, which is a potential considerable source of X-Risk from advanced AI systems.