SAFETY AWARE REINFORCEMENT LEARNING (SARL)

Anonymous authors
Paper under double-blind review

ABSTRACT

As reinforcement learning agents become increasingly integrated into complex, real-world environments, designing for safety becomes a critical consideration. We specifically focus on researching scenarios where agents can cause undesired side effects while executing a policy on a primary task. Since one can define multiple tasks for a given environment dynamics, there are two important challenges. First, we need to abstract the concept of safety that applies broadly to that environment independent of the specific task being executed. Second, we need a mechanism for the abstracted notion of safety to modulate the actions of agents executing different policies to minimize their side-effects. In this work, we propose Safety Aware Reinforcement Learning (SARL) – a framework where a virtual safe agent modulates the actions of a main reward-based task agent to minimize side effects. The safe agent learns a task-independent notion of safety for a given environment. The task agent is then trained with a regularization loss given by the distance between the native action probabilities of the two agents. Since the safe agent effectively abstracts a task-independent notion of safety via its action probabilities, it can be ported to modulate multiple policies solving different tasks across different environments without further training. We contrast this with solutions that rely on task-specific regularization metrics and test our framework on the SafeLife Suite, based on Conway's Game of Life, comprising a number of complex tasks in dynamic environments. We show that our solution is able to match the performance of solutions that rely on task-specific side-effect penalties on both the primary and safety objectives while additionally providing the benefit of generalizability and portability.

1 INTRODUCTION

Reinforcement learning (RL) algorithms have seen great research advances in recent years, both in theory and in their applications to concrete engineering problems. The application of RL algorithms extends to computer games (Mnih et al., 2013; Silver et al., 2017), robotics (Gu et al., 2017) and recently real-world engineering problems, such as microgrid optimization (Liu et al., 2018) and hardware design (Mirhoseini et al., 2020). As RL agents become increasingly prevalent in complex real-world applications, the notion of safety becomes increasingly important. Thus, safety related research in RL has also seen a significant surge in recent years (Zhang et al., 2020; Brown et al., 2020; Mell et al., 2019; Cheng et al.; Rahaman et al.).

1.1 SIDE EFFECTS IN REINFORCEMENT LEARNING ENVIRONMENTS

Our work focuses specifically on the problem of side effects, identified as a key topic in the area of safety in AI by Amodei et al. (2016). Here, an agent's actions to perform a task in its environment may cause undesired, and sometimes irreversible, changes in the environment. A major issue with measuring and investigating side effects is that it is challenging to define an appropriate side-effect metric, especially in a general fashion that can apply to many settings. The difficulty of quantifying side effects distinguishes this problem from safe exploration and traditional motion planning approaches that focus primarily on avoiding obstacles or a clearly defined failure state (Amodei et al., 2016; Zhu et al., 2020). As such, when learning a task in an unknown environment with complex dynamics, it is challenging to formulate an appropriate environment framework to jointly encapsulate the primary task and side effect problem.

Previous work on formulating a more precise definition of side effects includes work by Turner et al. (2019) on conservative utility preservation and by Krakovna et al. (2018) on relative reachability. These works investigated more abstract notions of measuring side effects based on an analysis of changes, reversible and irreversible, in the state space itself. While those works have made great progress on advancing towards a greater understanding of side effects, they have generally been limited to simple grid world environments where the RL problem can often be solved in a tabular way and value function estimations are often not prohibitively demanding. Our work focuses on expanding the concept of side effects to more complex environments, generated by the SafeLife suite (Wainwright and Eckersley, 2020), which provides more complex environment dynamics and tasks that cannot be solved in a tabular fashion. Turner et al. (2020) recently extended their approach to environments in the SafeLife suite, suggesting that attainable utility preservation can be used as an alternative to the SafeLife side metric described in Wainwright and Eckersley (2020) and Section 2. The primary differentiating feature of SARL is that it is metric agnostic, for both the reward and side effect measure, making it orthogonal and complimentary to the work by Turner et al. (2020).

In this paper, we make the following contributions which, to the best of our knowledge, are novel additions to the growing field of research in RL safety:

- *SARL*: a flexible, metric agnostic RL framework that can modulate the actions of a trained RL agent to trade off between task performance and a safety objective. We utilize the distance between the action probability distributions of two policies as a form of regularization during training.
- A generalizeable notion of safety that allows us to train a safe agent independent of specific tasks in an environment and port it across multiple complex tasks in that environment.

We provide a description of the SafeLife suite in Section 2, a detailed description of our method in Section 3, our experiments and results for various environments in Section 4 and Section 5 respectively, as well as a discussion in Section 6.

2 THE SAFELIFE ENVIRONMENT

The SafeLife suite (Wainwright and Eckersley, 2020) creates complex environments of systems of cellular automata based on a set of rules from Conway's Game of Life (Gardner, 1970) that govern the interactions between, and the state (alive or dead) of, different cells:

- any dead cell with exactly three living neighbors becomes alive;
- any live cell with less than two or more than three neighbors dies (as if by under- or overpopulation); and
- every other cell retains its prior state.

In addition to the basic rules, SafeLife enables the creation of complex, procedurally generated environments through special cells, such as a spawner that can create new cells and dynamically generated patterns. The agent can generally perform three tasks: *navigation*, *prune* and *append* which are illustrated in Figure 1 taken from Wainwright and Eckersley (2020).

The flexibility of SafeLife enables the creation of still environments, where the cell patterns do not change over time without agent interference, and dynamic environments, where the cell patterns do change over time without agent interference. The dynamic environments create an additional layer of difficulty, as the agent now needs to learn to distinguish between variations in the environment that are triggered by its own actions versus those that are caused by the dynamic rules independent of its actions. As described in Section 4, our experiments focus on the prune and append tasks in still and dynamic environments: *prune-still*, *prune-dynamic*, *append-still*, *append-dynamic*.

2.1 SAFELIFE SIDE EFFECT METRIC

There are two separate side effect metrics that we use in this paper: a training-time side effect metric that can easily be calculated at every frame, and a separate end-of-episode side effect metric used to validate overall agent safety. These metrics are identical to the side effect metrics described in Wainwright and Eckersley (2020).

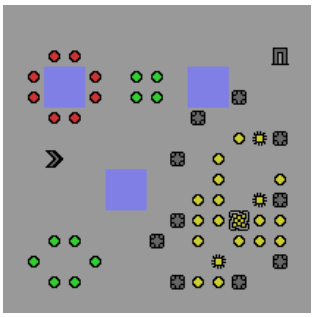


Figure 1: A simple level of the *SafeLife* environment containing an agent (»), a spawner (▩), crates (▣), and cells of life. The agent's goal is to remove unwanted red cells (*prune task*) and to create new patterns of life in the blue squares (*append task*). Once the agent has satisfactorily completed its goals it can leave via the level exit (▥). Note that all level boundaries wrap; they have toroidal topology.

The side effect metric used in training is found by comparing any given state to the starting state of that episode. Every cell in the grid that is different from the corresponding cell in the starting state (excluding differences due to agent movement or goal completion) is marked as having a side effect of +1. The total side effect of a state is the sum of side effects of individual cells. The difference in side effects between subsequent states is then to train safe agents as described in Section 3.

At the end of each episode, we perform a separate side effect calculation that is more robust to stochastic and chaotic dynamics observed in SafeLife. We prepare a counterfactual inaction baseline state at time t_{end}, defined as the state that would have been achieved had the agent followed an inaction policy for t_{end} time steps. We then evolve both the primary state and the inaction state for an additional $n = 1000$ time steps and store the average occupancy of green life cells at each point in the grid, producing two distributions. The total episodic side effect is then given the earth-mover distance between the two distributions using a modified Manhattan distance metric as described in Wainwright and Eckersley (2020). The *normalized* episodic side effect metric, shown in the figures in Section 5, is defined as the total episodic side effect divided by the number of green cells present at the start of the episode, showing the percentage of structures in each episode that are disrupted.

3 METHOD

3.1 TRAINING FOR REGULARIZED SAFE RL AGENT

Our method relies on regularizing the loss function of the RL agent with the distance of the task agent, A_θ, from the virtual safe actor, Z_ψ, as shown in Figure 2.

More formally, the general objective of the task agent A_θ can be expressed as:

$$\mathcal{F}_{\mathcal{A}}(\theta) = \mathcal{L}_\theta + \beta * \mathcal{L}_{\text{dist}}(\mathbb{P}_{\pi_\theta}, \mathbb{P}_{\pi_\psi}) \tag{1}$$

where β is a regularization hyperparameter, $\mathbb{P}_{\pi_\theta}$ represents the probability of taking a given action given by A_θ, and $\mathbb{P}_{\pi_\psi}$ represents the probability distribution of taking a given action according to Z_ψ. As shown in Equation 1, the actor loss $\mathcal{L}_\theta$ is regularized by the distance between the actions suggested by the task agent and the virtual safe agent. The gradient of the objective in Equation 1 expressed as the expectation of rewards of task agent actions α taken from a distribution of policies $\mathbb{P}_{\pi_\theta}$ is then given by:

$$\nabla_\theta \mathcal{F}_{\mathcal{A}}(\theta) = \nabla_\theta \mathbb{E}_{\alpha \sim \mathbb{P}_{\pi_\theta}}[\mathcal{L}(\theta)] + \beta * \nabla_\theta \mathbb{E}_{\alpha \sim \mathbb{P}_{\pi_\theta}, \zeta \sim \mathbb{P}_{\pi_\psi}}[\mathcal{L}_{\text{dist}}(\alpha, \zeta)] \tag{2}$$

where ∇_θ is independent from the virtual safe agent actions ζ given that Z_ψ is only dependent on ψ. This formulation enables training Z_ψ independently from A_θ, thereby abstracting the notion of safety away from the task. The gradient formulation underscores the importance for a distance metric $\mathcal{L}_{\text{dist}}$ that is differentiable to ensure that gradients update the task agent parameters θ from both terms of the augmented loss functions.

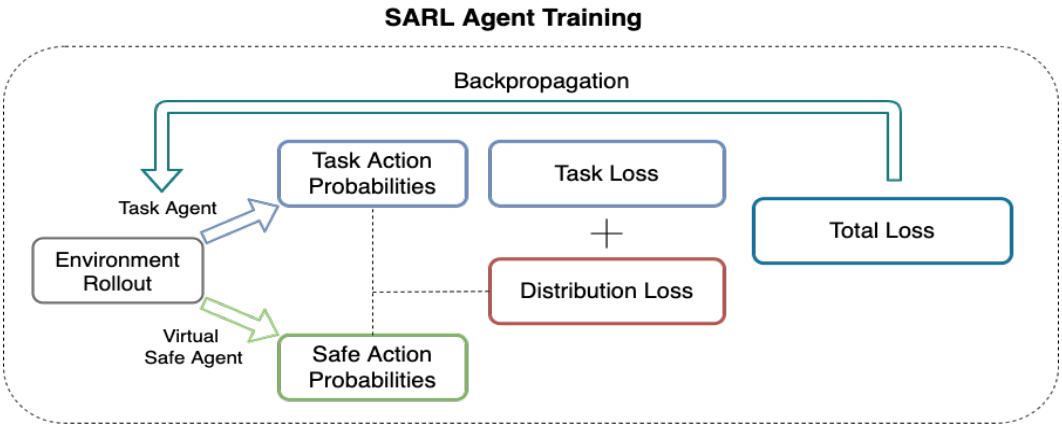


Figure 2: A co-training framework for safety aware RL training. The task agent, A_θ, determines the action taken in the environment and the resulting trajectories. The virtual safe agent, Z_ψ, receives the same state as A_θ and makes a suggestion for a safe action given the state. The distance between the action probabilities of the A_θ and Z_ψ is captured in the Distribution Loss, $\mathcal{L}_{\text{dist}}$. Z_ψ learns how to maximize the safety objective on its own set of environments in parallel to A_θ.

3.2 Distance Metrics for Loss Regularization

The primary objective of the regularization term is to express a notion of distance between a purely reward based action and a purely safety motivated action, thereby penalizing A_θ for taking a purely reward motivated action. We model the regularization term as the distance between probability distributions $\mathbb{P}_{\pi_\theta}$ and $\mathbb{P}_{\pi_\psi}$.

Given this formulation, previous work (Nowozin et al., 2016; Arjovsky et al., 2017; Huszár, 2015) has provided a number of choices for distance metrics in supervised learning problems with various advantages or shortfalls. One common method of measuring the difference in probability distributions is the KL Divergence, $D_{KL}(p\|q) = \int_x p(x) \log \frac{p(x)}{q(x)} dx$, where p and q are probability distributions described by probability density functions.

The KL Divergence, however, has some significant disadvantages – the most significant one being that the KL Divergence is unbounded when probability density functions to express the underlying distribution cannot be easily described by the model manifold (Arjovsky et al., 2017). Furthermore, the KL divergence is not symmetric given that $D_{KL}(p\|q) \neq D_{KL}(q\|p)$, and also does not satisfy the triangle inequality. One alternative to the KL Divergence is the Jensen-Shannon distance $D_{JS}(p\|q) = \frac{1}{2}D_{KL}(p\|m) + \frac{1}{2}D_{KL}(q\|m)$ with $m = \frac{1}{2}(p+q)$, which is symmetric, satisfies the triangle inequality and is bounded: $0 \leq D_{JS} \leq \log(2)$. These advantages make D_{JS} a good choice for the SARL algorithm, but as discussed extensively in Arjovsky et al. (2017), D_{JS} also has notable disadvantages, the most important being that D_{JS} is not guaranteed to always be continuous and differentiable in low manifold settings.

Another alternative to D_{JS} is the Wasserstein Distance. As discussed in Arjovsky et al. (2017), the Wasserstein Distance is generally better suited for calculating distances for low-dimensional manifolds compared to D_{JS} and other variants of the KL divergence. In its analytical form the Wasserstein Distance $W(p,q) = (\inf_{J \in \mathcal{J}(p,q)} \int \|x-y\|^p \mathrm{d}J(x,y))^{\frac{1}{p}}$, however, is intractable to compute in most cases leading many researchers to establish approximations of the metric. A common way of approximating the Wasserstein Distance is to re-formulate the calculation as an optimal transport problem of moving probability mass from p to q, as shown in Cuturi (2013) and Pacchiano et al. (2019). The dual formulation based on behavior embedding maps of policy characteristics described in Pacchiano et al. (2019) is particularly applicable for the SARL algorithm, leading us to adapt it as an additional alternative to the Jensen-Shannon Distance. In this formulation, policy characteristics are converted to distributions in a latent space of behavioral embeddings on which the Wasserstein Distance is then computed.

For our experiments in Section 4, we apply both D_{JS} and the dual formulation of the Wasserstein Distance described in Pacchiano et al. (2019) to compute the distance between $\mathbb{P}_{\pi_\theta}$ and $\mathbb{P}_{\pi_\psi}$.

3.3 Safety Aware Reinforcement Learning

The paragraphs above in Section 3 describe the individual components of SARL. The experiments outlined in Section 4 discuss SARL applied to Proximal Policy Optimization (PPO) (Schulman et al., 2017). Wainwright and Eckersley (2020) applied PPO for solving the different environments in the SafeLife suite, making SARL-PPO a natural extension. The loss formulation $\mathcal{L}_\theta^{PPO}$ used in Algorithm 1 is the same as the one decribed in Schulman et al. (2017):

$$\mathcal{L}_\theta^{PPO} = \mathbb{E}_t[\mathcal{L}_t^{Clip}(\theta) - c_1\mathcal{L}_t^{Value}(\theta) + c_2 S[\pi_\theta](s_t)] \tag{3}$$

As shown in more detail in Algorithm 1, A_θ is trained using the regularized loss objective described in Equation 1, while Z_ψ is trained exclusively on $\mathcal{L}_\theta^{PPO}$ using the frame-by-frame side effect information as the reward.

Algorithm 1 SARL - PPO

1: Initialize an actor $A_\theta(s)$ and virtual safety agent $Z_\psi(s)$ where s represents the state observation
2: Initialize a distance metric $\mathcal{L}_{\text{dist}}$
3: **while** training SARL-PPO **do**
4: **for** each actor update A_θ **do**
5: Run $A_\theta(s)$ to generate a minibatch of transitions α with task rewards r_{task}
6: Run $Z_\psi(s)$ to generate a minibatch of transitions ζ
7: Compute $\mathcal{L}_\theta^{PPO}$ and $\mathbb{P}_{\pi_\theta}$ using transitions in α
8: Compute $\mathbb{P}_{\pi_\psi}$ using transitions in ζ
9: Optimize A_θ using $\mathcal{L}_\theta^{PPO} + \beta * \mathcal{L}_{\text{dist}}(\mathbb{P}_{\pi_\theta}, \mathbb{P}_{\pi_\psi})$
10: **end for**
11: **for** each virtual agent update Z_ψ **do**
12: Run $Z_\psi(s)$ to generate a minibatch of transitions ζ with side-effect metric $r_{\text{side-effect}}$
13: Compute $\mathcal{L}_\psi^{PPO}$ using transitions in ζ
14: Optimize Z_ψ using $\mathcal{L}_\psi^{PPO}$ with $r_{\text{side-effect}}$ as the reward
15: **end for**
16: **end while**

The training algorithm is agnostic to the side effect metric, $r_{\text{side-effect}}$, used in the environment, leading itself to a plug-and-play approach where the virtual safe agent can modulate the task agent for a variety of different environment specific side effect metrics without major modification to the overall structure of the method.

In addition to training both A_θ and Z_ψ from scratch as shown in Algorithm 1, we also perform zero-shot generalization of a previously trained Z_ψ from a different environment to investigate whether the concept of side effects can be abstracted out of the environmental dynamics and the intricacies of the task. In this case, lines 11-15 from Algorithm 1 are not performed as no updates for Z_ψ are required, with Z_ψ only being used to modulate the behavior of A_θ via the distance metric regularization.

3.4 Tracking the Champion Policy

The SafeLife suite includes a complex set of procedurally generated environments, which can lead to a significant amount of variability throughout training and testing episodes. In order to account for this variability, we track the best policy throughout the training process for a fixed set of test levels for the different metrics we care about, specifically episode length, performance ratio and side effects, as described in Algorithm 2.

The champion policies operate on test levels, where no learning occurs, and track test-level metrics. This is particularly relevant to the side effect metric, described in Section 2, where we track the episodic side effect even though training occurs with frame-by-frame impact measure. We describe these metrics in greater detail in Section 4.

Algorithm 2 Champion Policy Tracking

Initialize Training and Champion Policy C_θ
for Every k Environment Steps **do**
    Evaluate task agent A_θ^k on fixed set of test levels
    if $Score_{A_\theta^k} > Score_{C_\theta}$ **then**
        $C_\theta = A_\theta^k$
    end if
end for

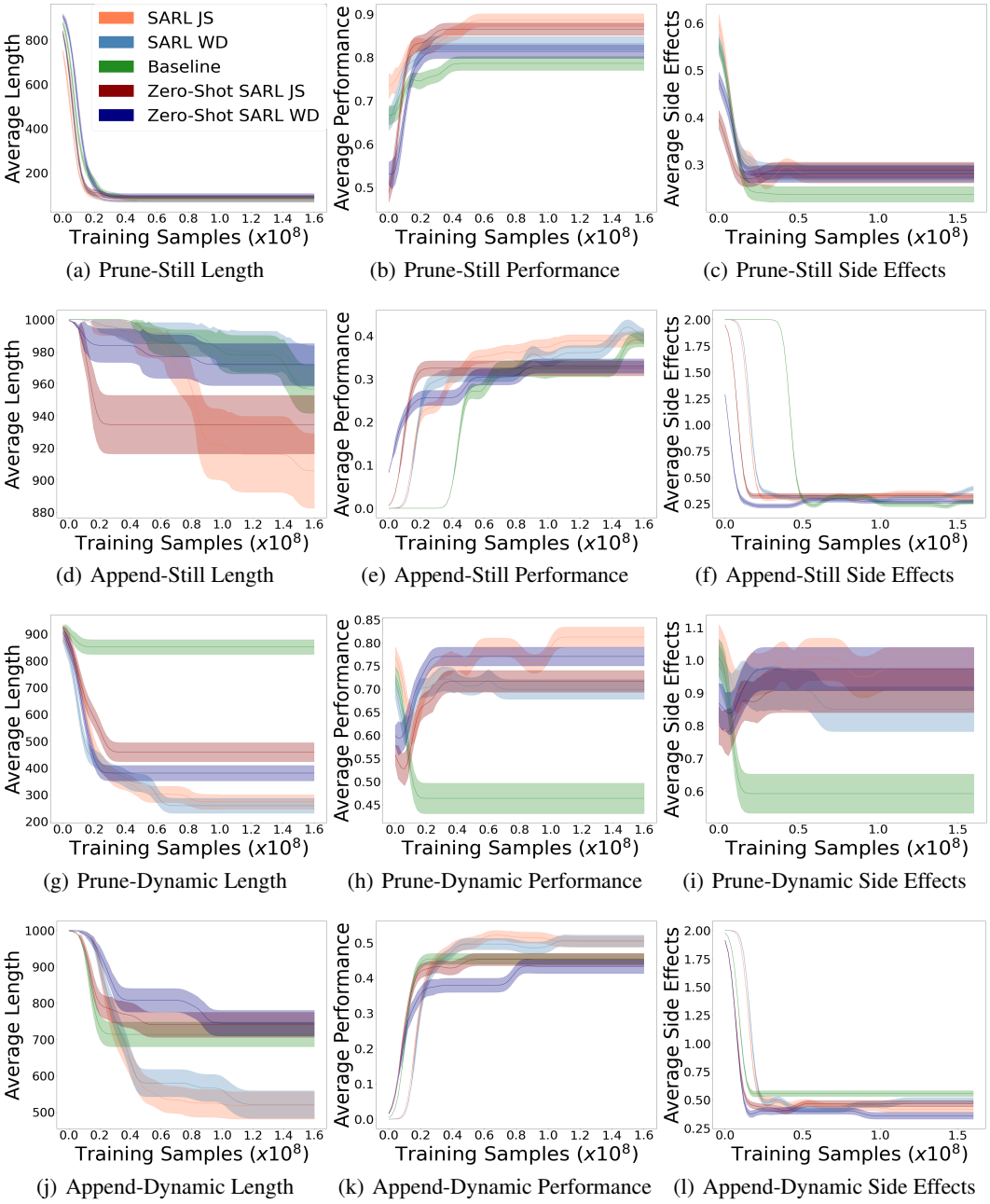
4 EXPERIMENTS

Our experiments contain the following algorithmic runs, where we track the results for the task agent A_θ:

- The reward-penalty baseline method described in (Wainwright and Eckersley, 2020) where the impact penalty of a given action is subtracted from the reward the agent receives for that particular frame.
- SARL agents where both the actor A_θ and virtual safety agent Z_ψ are training from scratch on the same environment using the Jensen-Shannon Distance as well as the dual formulation of the Wasserstein Distance described in (Pacchiano et al., 2019)
- SARL agents where A_θ is trained while Z_ψ is taken zero-shot from a previous training run on a different environment. The main purpose of this experiment is to show that the concept of side-effects in the SafeLife suite can be abstracted from the specific task (prune vs append) and the specific environment setting (still vs dynamic). The ability to extract a notion of side effects that does not rely on environmental signal for every frame enables us to train the virtual safety agent Z_ψ only once, usually on the simplest task, which can then be used to influence any agent on any subsequent task.

We conduct our experiments on four different tasks in the SafeLife suite: prune-still, append-still, prune-dynamic, append-dynamic. As described in Section 2, dynamic environments have natural variation independent of the actions of the agents, while all changes in still environments can be traced back to the actions of the agent. We evaluate our champion policies C_θ for the task agent every 100,000 environment steps on the episode length across a set of 100 different testing environments whose configurations are not part of the configurations of environments used in the training process. The length of an episode is the number of steps the agent takes to complete an episode, where a shorter length indicates that the agent can solve the task better and more efficiently. In our results in Section 5 we show the standard error of the champion measured in performance given by the ratio $\frac{\text{agent reward}}{\text{possible reward}}$ and the cumulative side effect measure described in Section 2 and Wainwright and Eckersley (2020).

The experimental results have a strong dependency on hyperparameters chosen, specifically the impact penalty fraction in the reward penalty baseline baseline and the regularization parameter β in SARL. Changing these parameters generally results in non-linear trade-offs between episode length, performance and side effects, meaning that policies with high performance often have high side effects and policies with low side effects often have low performance. In the cases of low side effects and low performance, the agent does not perform any significant actions that would either negatively (side effect) or positively (reward) disturb the environment. Our ideal goal is to have a policy that is both performant on the task and has low side effects. As such, in Section 5 we describe results of experiments that in our best judgement represent the best cases of such policies, and apply the same regularization hyperparameters across all environments. The full set of our algorithmic and regularization hyperparameters are shown in Appendix A. As discussed in more detail in Section 6, for future work we aim, and encourage others, to obtain Pareto optimal frontiers that describe the trade-off for the regularization hyperparameters more thoroughly.



(a) Prune-Still Length (b) Prune-Still Performance (c) Prune-Still Side Effects

(d) Append-Still Length (e) Append-Still Performance (f) Append-Still Side Effects

(g) Prune-Dynamic Length (h) Prune-Dynamic Performance (i) Prune-Dynamic Side Effects

(j) Append-Dynamic Length (k) Append-Dynamic Performance (l) Append-Dynamic Side Effects

Figure 3: Length Champion for SafeLife Suite of *1. prune-still (a-c), 2. append-still (d-f), 3. prune-dynamic (g-i), 4. append-dynamic (j-l)* tasks evaluated for 100 testing environments every 100,000 steps on *Episode Length (left column)* where shorter is better, *Performance Ratio (middle column)* where higher is better, and *Episodic Side Effect (right column)* where lower is better

5 RESULTS

The results of the experiments shown in Figure 3 demonstrate that a virtual safety agent trained on one task in the SARL framework can generalize zero-shot to other tasks and environment settings in the SafeLife suite, while maintaining competitive task and side effect scores compared to the baseline method. This allows us to abstract the notion of safety away from the environment specific side effect metric, and also increase the overall sample efficiency of the SARL method for subsequent training

runs. The SARL methods that are trained from scratch also show competitive task and side effect scores compared to the baseline method.

In the still environment we chose to generalize the virtual safety agents from *prune-still* and *append-still* to the other tasks using both distance metrics. The results show that zero-shot generalization of Z_ψ matches the behavior of SARL trained from scratch, as well as matching or outperforming the baseline method on episode length and performance.

Prune-Still Environment: The reward penalty baseline matches the episode length of all other methods while maintaining a slightly lower performance and side effects. All SARL methods, including both metrics and zero-shot SARL, generally perform equally well on length and side effects while the $SARL_{D_{JS}}$ has a slightly better performance than $SARL_{D_{WD}}$.

Append-Still Environment: The reward penalty baseline generally matches the performance and side effects of the SARL methods, while slightly underperforming $SARL_{D_{JS}}$ on episode length. $SARL_{D_{JS}}$ generally performs better on episode length compared to the other methods, both in training from scratch and the zero-shot experiments.

For the dynamic environment we chose to generalize the virtual safety agents trained on *prune-still* and *append-still* to the dynamic environment tasks using both distance metrics. In the zero-shot experiments, we applied the version of Z_ψ that is furthest away from the given setting, meaning *prune-still* is generalized to *append-dynamic* and *append-still* is generalized to *prune-dynamic*. The results show that zero-shot generalization of Z_ψ matches the behavior of SARL trained from scratch, as well as matching or outperforming the baseline method on some metrics.

Prune-Dynamic Environment: In this environment, we observe that the baseline method cannot solve the task, as shown by the fact that the episode length does not decrease significantly. However, it incurs very little side-effect cost. This indicates that the baseline agent is acting safer by not doing much in the environment, but actually fails to solve the primary task. All SARL agents outperform the baseline on episode length and performance ratio, indicating that SARL effectively learns the task.

Append-Dynamic Environment: The reward penalty baseline generally matches the behavior of the zero-shot SARL methods on episode length, performance and side effects. The SARL method trained from scratch, both $SARL_{D_{JS}}$ and $SARL_{D_{WD}}$ outperform the baseline as well as the zero-shot experiments on episode length and slightly on performance

6 Discussion

In this work, we explored the prospect of regularizing the loss function of an RL agent using distance metrics that encapsulate a notion of safe behavior for the RL agent. We believe this work shows the promise of this approach to train RL agents in environments where side effects are important. As mentioned in Section 1, side effects are often difficult to define, especially when interwoven with the primary task, and therefore measuring and interpreting side effects is an ongoing area of research. In order for our framework to be easily adoptable, we designed it to be flexible to different side effect metrics.

The idea of using suitable distance metrics to perform co-training of multiple RL agents has a variety of future research directions. One such avenue is the development of new distance metrics, including different variations of the Wasserstein Distance, as well as ones that can exploit various channels of information that we did not consider in out work (Parker-Holder et al., 2020). The ideal distance metric would capture both the information richness from the different channels and encode a notion of a safety objective which can then be transferred to the primary agent to influence its behavior.

There also exists a great opportunity to apply techniques from multi-objective optimization to side effect problem. The literature is rich with multi-objective optimization problems in supervised learning (Ma et al., 2020) (Sener, 2018) and reinforcement learning (Yang et al., 2019) (Xu et al., 2020) that show promising approaches to adopt a robust multi-objective framework to the side effect problem. The greatest promise of a multi-objective framework is the possibility of obtaining Pareto fronts (Yang et al., 2019) that describe the optimal trade-off between task performance and safety in a given environment, which would be immensely valuable to making decisions in real-world environments.

REFERENCES

D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Manè. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. 2017. URL http://arxiv.org/abs/1701.07875.

D. S. Brown, R. Coleman, R. Srinivasan, and S. Niekum. Safe Imitation Learning via Fast Bayesian Reward Inference from Preferences. 2020.

R. Cheng, R. M. Murray, and J. W. Burdick. Safety-Critical Continuous Control Tasks.

M. Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in Neural Information Processing Systems*, pages 1–9, 2013. ISSN 10495258.

M. Gardner. The fantastic combinations of jhon conway's new solitaire game'life. *Sc. Am.*, 223: 20–123, 1970.

S. Gu, E. Holly, T. Lillicrap, and S. Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3389–3396. IEEE, 2017.

F. Huszár. How (not) to Train your Generative Model: Scheduled Sampling, Likelihood, Adversary? pages 1–9, 2015. URL http://arxiv.org/abs/1511.05101.

V. Krakovna, L. Orseau, R. Kumar, M. Martic, and S. Legg. Penalizing side effects using stepwise relative reachability. *arXiv preprint arXiv:1806.01186*, 2018.

W. Liu, P. Zhuang, H. Liang, J. Peng, and Z. Huang. Distributed economic dispatch in microgrids based on cooperative reinforcement learning. *IEEE transactions on neural networks and learning systems*, 29(6):2192–2203, 2018.

P. Ma, T. Du, and W. Matusik. Efficient Continuous Pareto Exploration in Multi-Task Learning. 2020. URL http://arxiv.org/abs/2006.16434.

S. Mell, O. Brown, J. Goodwin, and S.-h. Son. Safe Predictors for Enforcing Input-Output Specifications. pages 1–10, 2019.

A. Mirhoseini, A. Goldie, M. Yazgan, J. Jiang, E. Songhori, S. Wang, Y.-J. Lee, E. Johnson, O. Pathak, S. Bae, et al. Chip placement with deep reinforcement learning. *arXiv preprint arXiv:2004.10746*, 2020.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

S. Nowozin, B. Cseke, and R. Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Advances in neural information processing systems*, pages 271–279, 2016.

A. Pacchiano, J. Parker-Holder, Y. Tang, A. Choromanska, K. Choromanski, and M. I. Jordan. Learning to Score Behaviors for Guided Policy Optimization. 2019. URL http://arxiv.org/abs/1906.04349.

J. Parker-Holder, A. Pacchiano, K. Choromanski, and S. Roberts. Effective diversity in population-based reinforcement learning. *arXiv preprint arXiv:2002.00632*, 2020.

N. Rahaman, S. Wolf, A. Goyal, R. Remme, and Y. Bengio. Learning the Arrow of Time. (1892): 1–19.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

O. Sener. Multi-Task Learning as Multi-Objective Optimization. (Nips), 2018.

D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

A. M. Turner, D. Hadfield-Menell, and P. Tadepalli. Conservative agency. *arXiv preprint arXiv:1902.09725*, 2019.

A. M. Turner, N. Ratzlaff, and P. Tadepalli. Avoiding side effects in complex environments. *arXiv preprint arXiv:2006.06547*, 2020.

C. L. Wainwright and P. Eckersley. SafeLife 1.0: Exploring side effects in complex environments. *CEUR Workshop Proceedings*, 2560:117–127, 2020. ISSN 16130073.

J. Xu, Y. Tian, P. Ma, D. Rus, S. Sueda, and W. Matusik. Prediction-Guided Multi-Objective Reinforcement Learning for Continuous Robot Control. 2020.

R. Yang, X. Sun, and K. Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. In *Advances in Neural Information Processing Systems*, 2019.

J. Zhang, B. Cheung, C. Finn, S. Levine, and D. Jayaraman. Cautious Adaptation For Reinforcement Learning in Safety-Critical Settings. 2020.

H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine. The ingredients of real-world robotic reinforcement learning, 2020.

A IMPLEMENTATION DETAILS

Hyperparameter	Value
γ	0.97
Learning Rate	$3e^{-4}$
Batch Size	64
Epochs per Training Batch	3
Environment Steps per Training Iteration	20
PPO Entropy Weight	0.01
PPO Entropy Clip	1.0
PPO Value Loss Coefficient	0.5
PPO Value Loss Clip	0.2
PPO Policy Loss Clip	0.2

Table 1: Hyperparameters for PPO-SARL

Environment	Baseline	SARL JS	SARL WD	SARL Zero-Shot JS	SARL Zero-Shot WD
Prune-Still	0.3	0.01	0.01	0.005	0.005
Append-Still	0.3	0.01	0.01	0.005	0.005
Prune-Dynamic	0.3	0.01	0.01	0.005	0.005
Append-Dynamic	0.3	0.01	0.01	0.005	0.005

Table 2: Hyperparameters for Side Effect Procedures – Baseline: Impact Penalty; SARL: β

B SAFELIFE TASKS

In this paper, we tackle four different tasks in the SafeLife environment:

- *Prune-Still*: In this task, the agent removes red cells placed at different locations throughout the environment. Given the still nature of the environment, there no variation occurring in the environment other than the ones caused by the actions of the agent.
- *Append-Still*: In this task, the agent fills in pattern specified by blue placed at different locations throughout the environment. Given the still nature of the environment, there no variation occurring in the environment other than the ones caused by the actions of the agent.
- *Prune-Dynamic*: In this task, the agent removes red cells placed at different locations throughout the environment. The dynamic nature of the environment causes inherent stochasticity of cell patterns that the agent interacts with. This inherent stochastic variation makes completing the task more difficult for the agent, as the distinction between variation caused by the agent's actions and environmentally driven variation is often inherently linked.
- *Append-Dynamic*: In this task, the agent fills in pattern specified by blue placed at different locations throughout the environment. The dynamic nature of the environment causes inherent stochasticity of cell patterns that the agent interacts with. This inherent stochastic variation makes completing the task more difficult for the agent, as the distinction between variation caused by the agent's actions and environmentally driven variation is often inherently linked.