Non-Robust Feature Mapping in Deep Reinforcement Learning

Ezgi Korkmaz [1]

Abstract

Adversarial perturbations to state observations can dramatically degrade the performance of deep reinforcement learning policies, and thus raise concerns regarding the robustness of deep reinforcement learning agents. A sizeable body of work has focused on addressing the robustness problem in deep reinforcement learning, and there are several recent proposals for adversarial training methods in the deep reinforcement learning domain. In our work we focus on the robustness of state-of-the-art adversarially trained deep reinforcement learning policies and vanilla trained deep reinforcement learning polices. We propose two novel algorithms to map non-robust features in deep reinforcement learning policies. We conduct several experiments in the Arcade Learning Environment (ALE), and with our proposed feature mapping algorithms we show that while the state-of-the-art adversarial training method eliminates a certain set of non-robust features, a new set of non-robust features more intrinsic to the adversarial training are created. Our results lay out concerns that arise when using existing state-of-the-art adversarial training methods, and we believe our proposed feature mapping algorithm can aid in the process of building more robust deep reinforcement learning policies.

1. Introduction

Learning reasonable representations from high dimensional raw data became possible with the introduction of deep neural networks. Recent successes of deep neural networks made several complex tasks achievable in many domains e.g. image recognition Sutskever et al. (2014), natural language processing (Hannun et al., 2014), self learning systems (Mnih et al., 2015). In particular, the utilization of deep neural networks as function approximators in reinforcement learning made it possible to learn policies for large state or action space Markov Decision Processes (MDP)s.

While the success of deep neural networks has grown several concerns have been raised related to robustness in the presence of specifically crafted adversarial perturbations (Goodfellow et al., 2015; Szegedy et al., 2014; Ilyas et al., 2019). To address the sensitivity problem several works have focused on making deep neural classifiers more robust to these specific perturbations by training with inputs modified via these perturbations (Madry et al., 2018; Goodfellow et al., 2015).

Several studies have demonstrated that the lack of robustness to adversarial perturbations observed in deep neural classifiers is also present in deep reinforcement learning policies (Huang et al., 2017; Kos & Song, 2017; Korkmaz, 2020). These concerns about the resilience of deep reinforcement learning were targeted by a line of work focused on building robust policies (Pinto et al., 2017; Pattanaik et al., 2018; Zhang et al., 2020). In our paper our aim is to try to answer the following questions: (i) Are there any vulnerability differences between the state-of-the-art adversarially trained deep reinforcement learning policies and vanilla trained deep neural policies?, and (ii) Do non-robust features still exist in different forms in state-of-the-art adversarially trained deep reinforcement learning policies? To fundamentally understand the sensitivities of deep reinforcement learning policies in our work we focus on adversarial training in deep reinforcement learning and make the following contributions:

- We propose two novel algorithms to visualize non-robust features in deep reinforcement learning policies.
- We conduct several experiments in the Arcade Learning Environment (ALE) with vanilla and adversarially trained deep reinforcement learning policies.
- We show that the state-of-the-art adversarial training method eliminates non-robust features intrinsic to the vanilla training techniques. However, we found that the state-of-the-art adversarial training method creates a new set of non-robust features intrinsic to adversarial training.

[1]Electrical Engineering and Computer Science School, KTH Royal Institute of Technology. Correspondence to: Ezgi Korkmaz <ezgikorkmazk@gmail.com>.

Proceedings of the 37th International Conference on Machine Learning, PMLR 108, 2021. Copyright 2021 by the author(s).

2. Background and Preliminaries

2.1. Reinforcement Learning

A Markov Decision Process (MDP) is represented as a tuple $(\mathcal{S}, \mathcal{A}, r, \mathcal{P}, \gamma, \rho_0)$ where $\mathcal{S}$ represents a finite set of states, $\mathcal{A}$ represents a finite set of actions, r represents the reward function, $\mathcal{P}$ represents the Markovian transition kernel, γ represents the discount factor and ρ_0 represents the initial state distribution. The aim is to learn a value for each state-action pair to determine the expected discounted cumulative rewards to be obtained if action a is executed in state s. This is achieved by learning the state-action value function $Q(s,a) = \mathbb{E}_{a\sim\pi(\cdot|s)}[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)|s_0 = s, a_0 = a]$. Once the state-action value function is learnt the optimal policy $\pi^*(a|s)$ is given by taking the action $a^*(s) = \arg\max_a Q(s,a)$ in state s.

2.2. Adversarial Perturbations and Adversarial Training

Goodfellow et al. (2015) introduced a computationally inexpensive way to produce adversarial examples based on linearly approximating the loss function around a given input,

$$x_{\text{adv}} = x + \epsilon \cdot \frac{\nabla_x J(x,y)}{||\nabla_x J(x,y)||_p}, \tag{1}$$

where x is a given input, y is the output labels, and $J(x,y)$ is the cost function used to train the network. Note that perturbations computed in this way are bounded by ϵ. Kurakin et al. (2016) focused on an iterative search using the fast gradient sign method introduced by Goodfellow et al. (2015).

$$x^0_{\text{adv}} = x, \tag{2}$$

$$x^{N+1}_{\text{adv}} = \text{clip}_\epsilon(x^N_{\text{adv}} + \alpha \text{sign}(\nabla_x J(x^N_{\text{adv}}, y))) \tag{3}$$

Madry et al. (2018) referred to this class of iterative methods as projected gradient descent (PGD), and gave a theoretical justification for training with inputs modified by such perturbations using robust optimization theory.

2.3. Robustness in Deep Reinforcement Learning

Investigation into the robustness of deep reinforcement learning policies was first conducted by Huang et al. (2017) and Kos & Song (2017). In these studies the authors show the lack of robustness of deep reinforcement learning policies to fast gradient sign method produced perturbations. Pattanaik et al. (2018) propose to increase the probability of worst possible action (i.e. $\arg\min_a Q(s,a)$) in a given state to achieve higher degradation of the agent's performance. Pinto et al. (2017) models the interaction between the adversary and the agent as a zero-sum Markov game and proposes a training strategy to compute a robust policy for the agent. More recently, Zhang et al. (2020) propose to model this problem as a State-Adversarial Markov Decision Process (SA-MDP). In this study, the authors claim that their proposed SA-MDP model and algorithm obtain theoretically justified robust policies towards both natural errors and adversarial perturbations.

3. Mapping Non-Robust Features in Deep Reinforcement Learning Policies

In this paper we aim to seek answers for the following questions:

- What are the susceptibility differences between state-of-the-art adversarially trained deep reinforcement learning policies and vanilla trained deep reinforcement learning policies?
- Do the sensitivities of deep reinforcement learning policies shifts from worst-case ℓ_p-norm bounded perturbations towards different directions in the input with adversarial training?
- Does adversarial training create a new set of non-robust features while eliminating the existing ones?

In this section we propose two different methods to visualize vulnerabilities of deep reinforcement learning policies to their input observations. First, we describe our proposed method of feature vulnerability mapping KMAP in detail. To be able to visualize weaknesses we record the drop in the state-action value $Q(s,a)$ caused by setting each pixel in s to zero one at a time. In particular, let $Z_{i,j} : \mathcal{S} \to \mathcal{S}$ be the function which sets the i,j coordinate of s to zero and leaves the other coordinates unchanged. We define,

$$\mathcal{K}(i,j) = Q(s,a^*) - Q(s, \arg\max_a Q(Z_{i,j}(s), a)). \tag{4}$$

Note that the difference in Equation 4 represents the drop in the Q-value in state s, when taking the optimal action for the state $Z_{i,j}(s)$. Therefore, $\mathcal{K}(i,j)$ aims to measure the drop in the Q-values of the reinforcement learning policy with respect to individual pixel changes. In other words, $\mathcal{K}(i,j)$ is a mapping of features to an importance metric determined by the deep reinforcement learning policy. We describe our proposed KMAP method in detail in Algorithm 2.

As a natural point of comparison we propose another algorithm HMAP to visualize input based vulnerabilities. In particular, HMAP is based on measuring the effect of each individual pixel on the decision of the deep reinforcement learning policy by measuring the cross-entropy loss between $\pi(s,a)$ and $\pi(Z_{i,j}(s), a)$.

$$\mathcal{H}(i,j) = -\sum_{a\in A} \pi(s,a)\log(\pi(Z_{i,j}(s), a)) \tag{5}$$

Algorithm 1 HMAP Feature vulnerability mapping

Input: State-action value function $Q(s, a)$, actions a, states s, policy $\pi(s, a)$, T_d size of the dimension d of the state s, and $s(i, j)$ is the value of the i, j-th pixel in state s.
Output: Visual weakness mapping function $\mathcal{H}(i, j)$
$s_{\text{aug}} = s$
for $i = 1$ **to** T_1 **do**
  for $j = 1$ **to** T_2 **do**
    $s_{\text{aug}}(i, j) = 0$
    $\pi(s, a) = \text{softmax}(Q(s, a))$
    $\pi(s_{\text{aug}}, a) = \text{softmax}(Q(s_{\text{aug}}, a))$
    $\mathcal{H}(i, j)$ +=$-\sum_{a \in A} \pi(s, a) \log(\pi(s_{\text{aug}}, a))$
    $s_{\text{aug}} = s$
  end for
end for
Return: $\mathcal{H}(i, j)$

Algorithm 2 KMAP Feature vulnerability mapping

Input: State-action value function $Q(s, a)$,actions a, states s, T_d size of the dimension d of the state s, and $s(i, j)$ is the value of the i, j-th pixel in state s.
Output: Visual weakness mapping function $\mathcal{K}(i, j)$
$s_{\text{aug}} = s$
for $i = 1$ **to** T_1 **do**
  for $j = 1$ **to** T_2 **do**
    $s_{\text{aug}}(i, j) = 0$
    a^*_{aug} = $\text{argmax}_a\ Q(s_{\text{aug}}, a)$
    a^* = $\text{argmax}_a\ Q(s, a)$
    $\mathcal{K}(i, j)$ += $Q(s, a) - Q(s, a^*_{\text{aug}})$
    $s_{\text{aug}} = s$
  end for
end for
Return: $\mathcal{K}(i, j)$

where we compute the policy $\pi(s, a)$ via the softmax of the state-action value function $Q(s, a)$,

$$\pi(s, a) = \frac{e^{Q(s,a)/T}}{\sum_{a \in A} e^{Q(s,a)/T}}. \tag{6}$$

Note that T represents the temperature constant. We describe the HMAP method in detail in Algorithm 1.

4. Results on KMAP and HMAP

The vanilla trained deep reinforcement learning policies are trained via Double Deep Q-Network (DDQN) Wang et al. (2016), and the the-state-of-the-art adversarially trained deep reinforcement learning policy is trained via State Adversarial Double Deep Q-Network (SA-DDQN) Zhang et al. (2020). Note that Zhang et al. (2020) uses a regularization term,

$$\mathcal{R}(\theta) = \sum_s \max\{\max_{s \in \mathcal{D}(s)} \max_{a \neq a^*} Q_\theta(s_{\text{adv}}, a) \tag{7}$$

$$- Q_\theta(s_{\text{adv}}, a^*(s)), c\} \tag{8}$$

added in the temporal difference loss in Q-learning in training time where $\mathcal{D}(s) := \{s_{\text{adv}} | \left\|s_{\text{adv}} - s\right\|_\infty \leq \epsilon\}$. The deep reinforcement learning policies are trained in Arcade Learning Environment (ALE) proposed by Bellemare et al. (2013) in OpenAI version Brockman et al. (2016).

Figure 2 and Figure 1 show heatmaps of feature vulnerability mapping KMAP $\mathcal{K}(i, j)$ and HMAP $\mathcal{H}(i, j)$ for the state-of-the-art adversarially trained deep reinforcement learning policy and vanilla trained deep reinforcement learning policy for Freeway. We observe that while the KMAP $\mathcal{K}(i, j)$ pattern for the vanilla trained agent lies on the portion of the input where the optimal policy is executed by the agent, the KMAP $\mathcal{K}(i, j)$ for the adversarially trained deep reinforcement learning policy has a straightforward grid pattern. Based on these results, we hypothesize that adversarial training decouples vulnerability from the features relevant to the optimal policy learned by the agent. The decoupling of relevant features and vulnerability can be seen as a way in which adversarial training shifts the vulnerabilities of deep reinforcement learning policies.

While visual observation indicates very different vulnerability patterns for these two disjoint training strategies, we also introduce a quantitative metric to compare the results of KMAP and HMAP for vanilla and adversarially trained agents. In particular, we use the ratio of the ℓ_1 and ℓ_2 norms to measure the sparsity via,

$$S(\mathcal{K}) = \frac{\|\mathcal{K}\|_1}{\|\mathcal{K}\|_2}. \tag{9}$$

Here smaller values of $S(\mathcal{K})$ correspond to sparser vulnerability patterns.

In Table 1 we show the sparsity results respectively for KMAP $\mathcal{K}(i, j)$ and HMAP $\mathcal{H}(i, j)$ for adversarially trained deep reinforcement learning policies and vanilla trained deep reinforcement learning policies. We observe that for KMAP the vulnerability of adversarially trained models with respect to features are more sparse than the vanilla trained agents. The results for HMAP are more mixed, and it is often barely possible to detect the sparsity difference via $S(\mathcal{H})$. In general, KMAP $\mathcal{K}(i, j)$ provides a better estimation of sensitivity of deep reinforcement learning policies to individual pixel changes than HMAP $\mathcal{H}(i, j)$. While KMAP $\mathcal{K}(i, j)$ captures the actual impact of the feature change on the decision of the deep reinforcement learning policy HMAP $\mathcal{H}(i, j)$ captures the difference between the softmax policy distributions $\pi(s, a)$ and $\pi(Z_{i,j}(s), a)$,

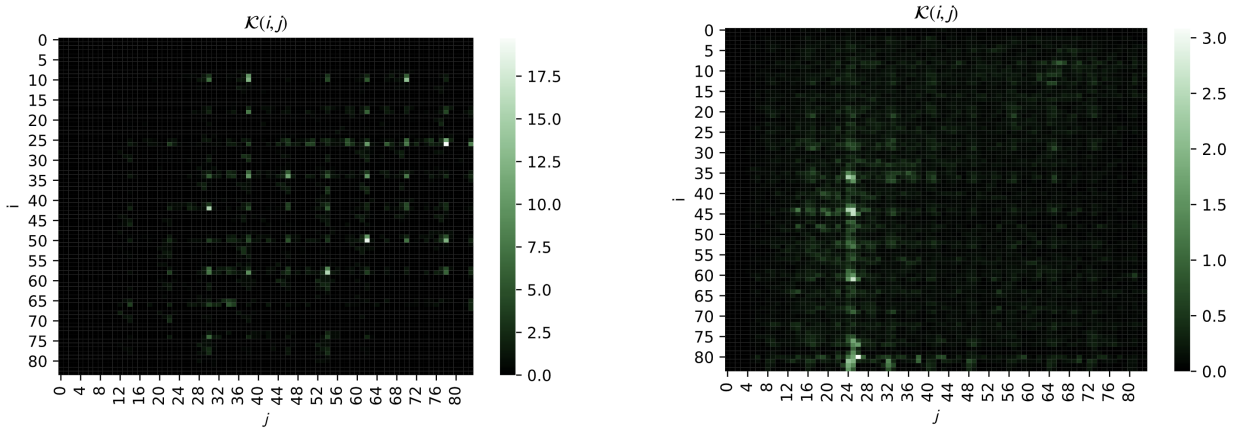

Figure 1. KMAP $\mathcal{K}(i,j)$ and HMAP $\mathcal{H}(i,j)$ heatmaps for state-of-the-art adversarially trained deep reinforcement learning policy and vanilla trained deep reinforcement learning policy in Freeway. Left: Adversarially trained (SA-DDQN). Right: Vanilla trained (DDQN).

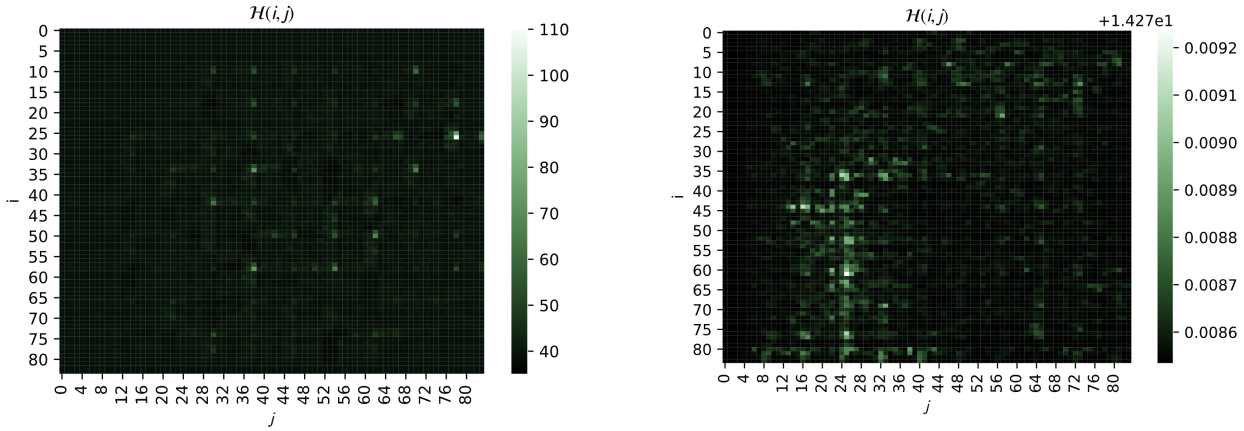

Figure 2. HMAP $\mathcal{H}(i,j)$ heatmaps for state-of-the-art adversarially trained deep reinforcement learning policy and vanilla trained deep reinforcement learning policy in Freeway. Left: Adversarially trained (SA-DDQN). Right: Vanilla trained (DDQN).

Table 1. Sparsity results of KMAP $\mathcal{K}(i,j)$ and HMAP $\mathcal{H}(i,j)$ for adversarially and vanilla trained deep reinforcement learning policies.

Training Method Sparsity	Vanilla Trained $S(\mathcal{K})$	Adversarially Trained $S(\mathcal{K})$	Vanilla Trained $S(\mathcal{H})$	Adversarially Trained $S(\mathcal{H})$
Freeway	53.7272	20.4641	83.9999	83.91587

which do not necessarily correspond to the decisions made by the neural policy.

5. Conclusion

In this paper we focused on investigating the vulnerabilities of deep reinforcement learning policies with respect to their inputs. We propose two different algorithms that we call KMAP and HMAP to detect vulnerabilities with respect to input in deep reinforcement learning policies. We compare the state-of-the-art adversarially trained neural policies and vanilla trained neural policies with our proposed methods KMAP and HMAP via several experiments in various environments. With the help of our proposed feature vulnerability mapping algorithm we found that while adversarial training removes sensitivity to certain features, it builds sensitivity towards a new set of features. We believe this work lays out the vulnerabilities of adversarially trained neural policies in a systematic way, and can be an initial step towards building robust and reliable deep reinforcement learning agents.

References

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research.*, pp. 253–279, 2013.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv:1606.01540*, 2016.

Goodfellow, I., Shelens, J., and Szegedy, C. Explaning and harnessing adversarial examples. *International Conference on Learning Representations*, 2015.

Hannun, A., Case, C., Casper, J., Catanzaro, B., Greg, D., Else, E., Prenger, R., Satheesh, S., Shubho, S., Coates, A., and Ng, A. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

Huang, S., Papernot, N., Goodfellow, Ian an Duan, Y., and Abbeel, P. Adversarial attacks on neural network policies. *Workshop Track of the 5th International Conference on Learning Representations*, 2017.

Ilyas, A., Santurkar, S., Tsipras, D., Engstrom, L., Tran, B., and Madry, A. Adversarial examples are not bugs, they are features. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 125–136, 2019.

Korkmaz, E. Nesterov momentum adversarial perturbations in the deep reinforcement learning domain. *International Conference on Machine Learning, ICML 2020, Inductive Biases, Invariances and Generalization in Reinforcement Learning Workshop.*, 2020.

Kos, J. and Song, D. Delving into adversarial attacks on deep policies. *International Conference on Learning Representations*, 2017.

Kurakin, A., Goodfellow, I., and Bengio, S. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, a. G., Graves, A., Riedmiller, M., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Pattanaik, A., Tang, Z., Liu, S., and Gautham, B. Robust deep reinforcement learning with adversarial attacks. *In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2040–2042, 2018.

Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. Robust adversarial reinforcement learning. *International Conference on Learning Representations ICLR*, 2017.

Sutskever, I., Vinyals, O., and Le, Q. V. . Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 2014.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *In Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., and De Freitas, N. Dueling network architectures for deep reinforcement learning. *Internation Conference on Machine Learning ICML.*, pp. 1995–2003, 2016.

Zhang, H., Chen, H., Xiao, C., Li, B., Liu, M., Boning, D. S., and Hsieh, C. Robust deep reinforcement learning against adversarial perturbations on state observations. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.