A ATTACK ALGORITHM

The detailed steps of our attack algorithm are presented in Algorithm 1. To preserve the universality, we keep the search problem independent from our algorithm because the selection may vary according to the environment. For the Overcooked environment, an effective approach is to first compute the perturbation significance of each unit perturbation, followed by obtaining legal combinations of the top unit perturbations, and finally selecting among those combinations.

Algorithm 1 Our attack algorithm

Input: The original MDP, victim policy π, number of needed adversarial states k, distance constraint ϵ, frequency threshold p_{freq}, and search algorithm $Search$

Output: k adversarial initial states

$\tau \leftarrow$ collect trajectories from MDP with policy π

$\mathcal{S}^E_{reachable} \leftarrow$ get the reachable environmental state space from MDP

$s^E_0 \leftarrow$ the initial environmental state of MDP

$\mathcal{S}^E_{feasible} \leftarrow$ states in $\mathcal{S}^E_{reachable}$ that distance no more than ϵ from s^E_0

$\widehat{\mathcal{S}^E} \leftarrow$ states in $S_{feasible}$ that consist of perturbations with appearance frequency less than p_{freq} in τ

$Dataset \leftarrow \{(s_t, a_t) | s_t \in \mathcal{S}_\tau, a_t = \pi_{opt}(s_t)\}$

$f(\hat{s}^E_0) \leftarrow \sum_{(s_t,a_t)\in Dataset} [\frac{\partial \pi(s_t,a_t)}{\partial s_t}(s^E_0 - \hat{s}^E_0)]$

$\{\hat{s}^E_1, \ldots, \hat{s}^E_k\} \leftarrow Search(f, \hat{\mathcal{S}^E}, k)$

Return: $\{\hat{s}_1, \ldots, \hat{s}_k\}$

B IMPLEMENTATION DETAILS

B.1 AGENT TRAINING

The policy architecture follows the same designation of Carroll et al. (2019), which is parameterized as a 3-layer CNN (5×5, 3×3, and 3×3 respectively) followed by a 3-layer MLP with a hidden size of 32. For both the CNN and the MLP, LeakyReLU is selected as the activation function. After the MLP, two separate linear layers are used for generating the action and estimating the value.

The specific hyperparameters for the PPO are outlined in Table 1. Most of them align with the setting on *Coordination Room* in PPO-BC (Carroll et al., 2019). Unlike Carroll et al. (2019), all layouts share the same setting of hyperparameters in our experiment. Our code is partially based on the implementation of Kostrikov (2018).

Table 1: PPO hyperparameters

Learning rate	0.001
Value Function coefficient	0.05
Reward shaping horizon	2.5e6
Minibatch size	2000
# Minibatches	6
Discount factor gamma	0.99
GAE lambda	0.98
Entropy coefficient	0.01
max gradient norm	0.1
Clip range	0.05

All the agents (SP, FCP, and FCP partners) share the same architecture and hyperparameters introduced above.

B.2 ATTACK

To perform the attack, we first collect 20 trajectories of the victim agent, each with 800 time steps. The p_{freq} is set to 0.03, *i.e.*, perturbations whose frequency of occurrence exceeds this threshold will be removed from the candidate. We then obtain the top k combined perturbations through Algorithm 1 as the adversarial outputs.

We run experiment on a *Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz* and a *NVIDIA GeForce RTX 3090* GPU. Attacking on a single agent takes 20 to 40 minutes, depending on the complexity of layouts.

B.3 DEFENSE

Similar to the attack, 20 trajectories are collected for the supervised learning stage, each with 800 time steps. The detailed settings are:

- The temperature T for $\mathcal{L}_p$ is set to 1.5.
- The tolerance on value estimation α for $\mathcal{L}_p$ is set to 0.05.
- The loss weight β for $\mathcal{L}$ is set to 1.
- We use 10 perturb initial states for the supervised learning, 5 are sampled from the attack results and 5 are random perturbations.
- We take out 30% of the data as the validation set.
- The supervised learning last 100 epochs.
- The learning rate is set to 0.001.
- The batch size is set to 8000.

For the fine-tuning stage, the distribution of the initial state is set as:

- 30% chance of selecting the standard initial state.
- 70% chance of sampling from the perturbed initial states.

Compared to the initial training of agents, the coefficient of the value function is adjusted to 0.01, and the reward shaping horizon is reduced to 2e6. The remaining hyperparameters remain unchanged.

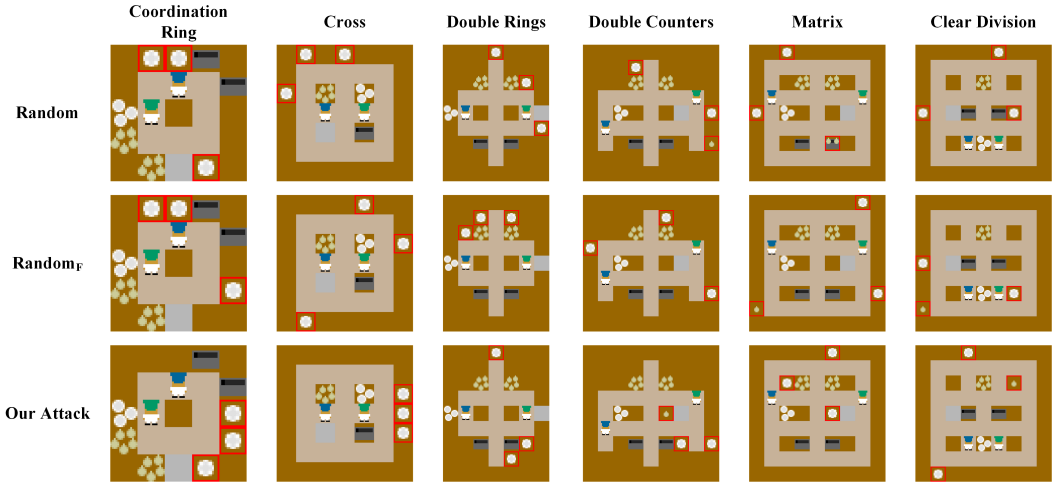
Using the same hardware as introduced above, the supervised learning of each agent takes approximately 15 minutes, while the fine-tuning stage takes about 2 to 4 hours with regard to the complexity of layouts.

C ATTACK VISUALIZATION

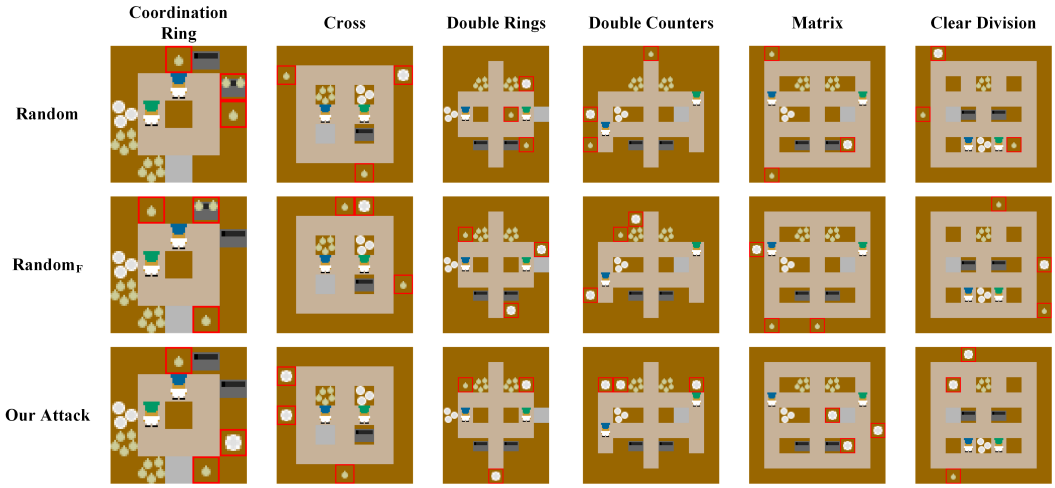
We visualize representative examples of the perturbed initial states generated by all methods in Figure 1 and Figure 2, where the perturbations are marked with red rectangles. Intuitively, it is hard to distinguish the adversarial examples from random perturbation, which implies the stealthiness of our attack. On the other hand, although the perturbations in all the adversarial initial states seem chaotic, the adversarial states generated by our attack can reduce the reward more significantly, which demonstrates the effectiveness of selecting perturbations by computing the perturbation significance.

Besides, the results suggest that the attack effect stems from the out-of-distribution perturbations. In comparison to the random baseline, random_F filters out states that the agent may have already encountered during the training process and achieves better performance. For example, putting ingredients in the pot which the agent usually uses will not incapacitate the agent but help it get a higher reward.

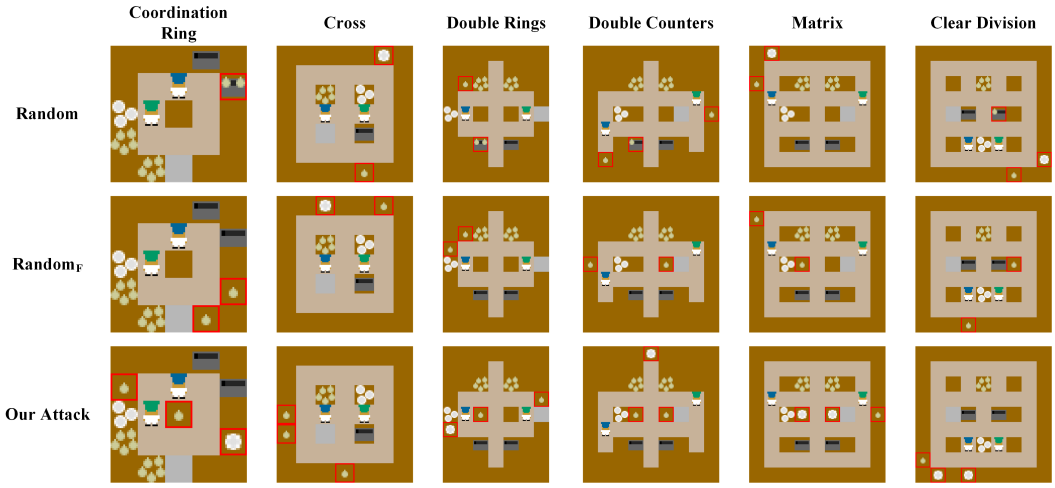
Another finding is that a larger perturbation budget ϵ may improve the attack performance. As the examples shown in Figure 1c and Figure 2c, the weakest perturbations that are randomly generated often do not spend the full budget, implying that more unit perturbations possibly lead to a stronger attack. Such a result suggests that most of the unit perturbations are harmful to the agent, which also reveals the fragility of commonly trained agents.



(a) Perturbed initial states that achieve the best attack performance

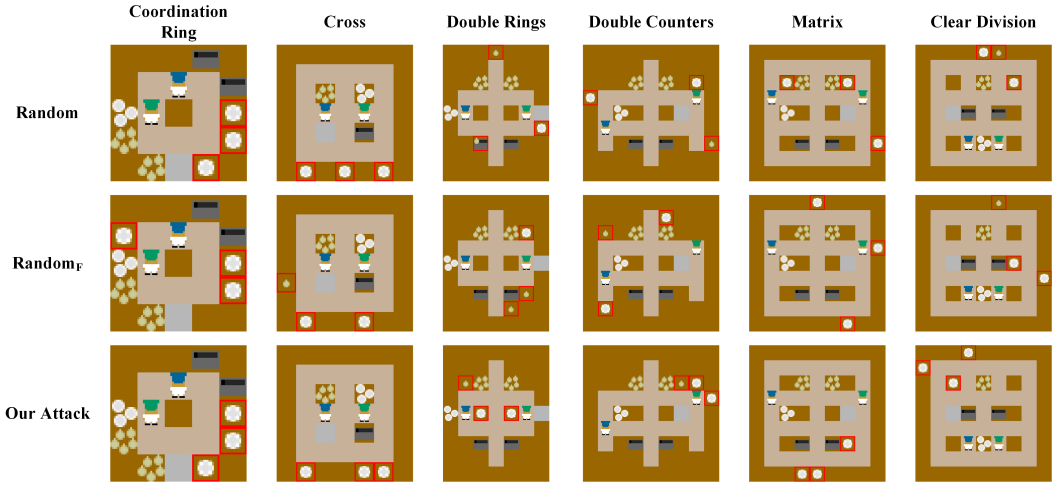


(b) Perturbed initial states that achieve the median attack performance

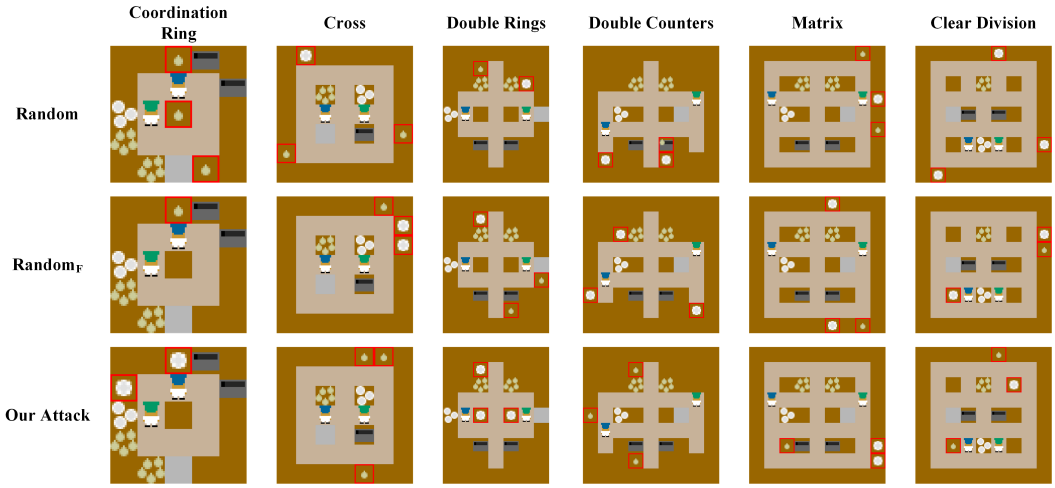


(c) Perturbed initial states that achieve the worst attack performance

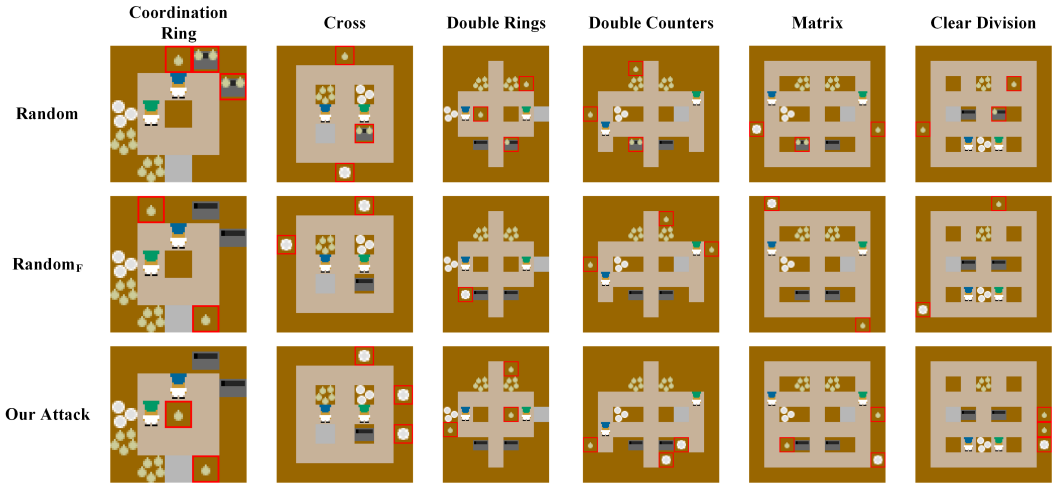
Figure 1: Perturbed initial states that target at SP agents



(a) Perturbed initial states that achieve the best attack performance



(b) Perturbed initial states that achieve the median attack performance



(c) Perturbed initial states that achieve the worst attack performance

Figure 2: Perturbed initial states that target at FCP agents

D ABLATION STUDY

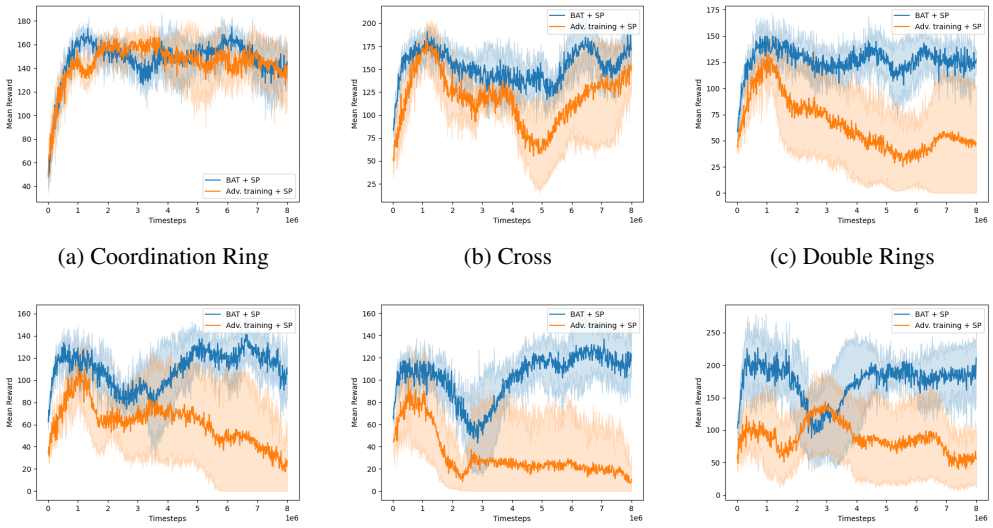
Both stages of our BAT are necessary. To study their impact, we conduct an ablation study to compare the BAT with two variants: (1) BAT without fine-tuning, denoted as **w.o. FT**; (2) BAT without supervised training, which is the naive adversarial training, denoted as **adv. training**. We run the experiments under the same setting as BAT and present the quantitative results in Table 2. We also show the training curves at the fine-tuning stage of our BAT and the naive adversarial training in Figure 3 and Figure 4. The rewards at each timestep are the mean value of 5 agents, and the shade denotes a 95% confidence interval. We emphasize that the rewards are calculated across the distribution of initial states, thus representing the overall performance of agents without distinguishing whether the rewards are obtained in the standard environment or the perturbed environments.

The results show that a lack of each stage will cause significant decreases in rewards. As shown in Table 2, although the agents without fine-tuning can defend against the perturbations, they fail to maintain their capability in the standard environment. The phenomenon is in line with expectations, since the supervised pre-training targets to reduce the effect of perturbations, rather than obtaining higher overall rewards.

As we briefly discussed in Section 3, naive adversarial training works in simple environments but may fail when the task is hard. Although it shows comparable performance to our BAT in the *Coordination Ring*, there exists a significant reduction of performance in other layouts. The reduction in performance shows a positive correlation with the complexity of layouts, which supports our motivation that adversarial training is not directly applicable to reinforcement learning. We also notice that FCP agents perform significantly better in complex layouts, substantiating that the adversarial training performance highly depends on the initial capability of agents, and ulteriorly supports the effectiveness and necessity of our supervised pre-training stage. Our two-stage framework can be useful not only in defending against environmental perturbations but also applicable to any adversarial training in reinforcement learning forms.

Table 2: Ablation results for variants of BAT (average scores with standard errors)

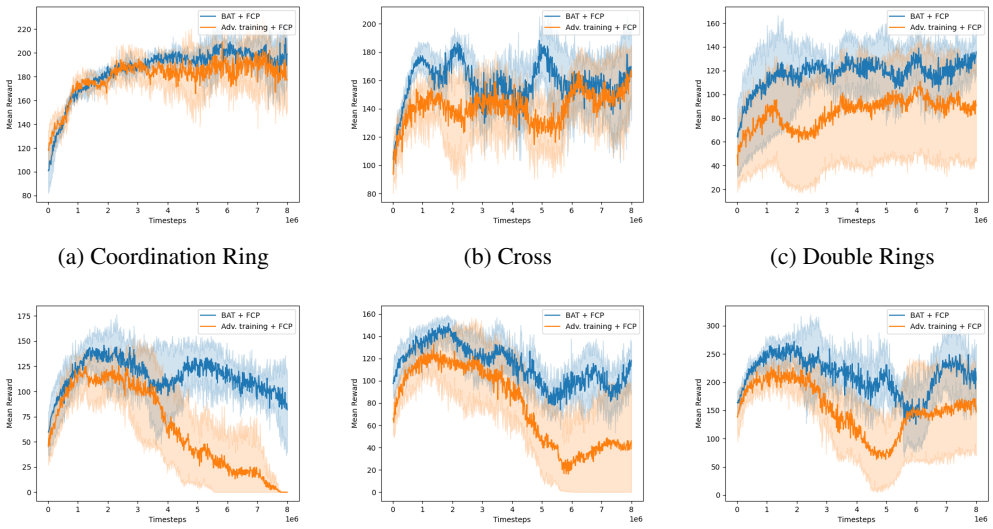
Method	Agent	Attack	Coord. Ring	Cross	Doub. Rings	Doub. Coun.	Matrix	Clear Div.
W.o. FT	SP	No attack	185.1 ± 24.7	267.5 ± 13.5	191.8 ± 7.8	207.8 ± 9.5	209.3 ± 15.3	269.9 ± 22.1
		Random	96.0 ± 7.1	105.7 ± 8.4	62.7 ± 5.4	66.7 ± 5.6	77.9 ± 6.9	97.0 ± 8.1
		Our attack	107.5 ± 5.1	64.9 ± 6.8	61.6 ± 5.5	56.2 ± 4.5	60.0 ± 4.5	76.7 ± 7.8
	FCP	No attack	265.6 ± 13.9	262.2 ± 13.3	179.5 ± 41.0	155.8 ± 26.1	223.2 ± 10.8	394.1 ± 17.4
		Random	239.3 ± 6.0	192.1 ± 6.2	123.2 ± 10.1	104.5 ± 7.9	163.7 ± 5.5	260.9 ± 13.5
		Our attack	227.4 ± 5.5	144.0 ± 5.7	99.5 ± 8.4	93.7 ± 7.1	**135.4 ± 7.0**	210.2 ± 15.0
Adv. training	SP	No attack	381.8 ± 19.7	434.2 ± 7.0	157.7 ± 76.3	264.2 ± 59.5	74.4 ± 66.4	202.6 ± 89.3
		Random	260.2 ± 14.9	224.2 ± 16.2	82.0 ± 14.9	93.7 ± 13.3	47.1 ± 13.8	108.6 ± 18.7
		Our attack	219.7 ± 14.2	172.1 ± 14.3	66.7 ± 12.1	35.6 ± 5.9	12.1 ± 3.7	7.0 ± 1.3
	FCP	No attack	**506.2 ± 23.6**	421.1 ± 15.7	274.8 ± 63.1	225.9 ± 73.8	299.7 ± 24.4	546.6 ± 65.0
		Random	361.4 ± 16.7	234.4 ± 15.3	138.1 ± 15.6	58.6 ± 10.8	105.1 ± 12.5	309.6 ± 27.9
		Our attack	210.4 ± 21.6	169.7 ± 14.2	106.4 ± 14.6	7.0 ± 1.7	58.3 ± 10.8	198.3 ± 21.7
BAT	SP	No attack	372.8 ± 9.8	459.7 ± 14.3	**373.3 ± 24.5**	**352.8 ± 4.5**	335.9 ± 23.4	612.2 ± 35.6
		Random	269.6 ± 15.1	279.4 ± 14.7	190.2 ± 13.0	180.6 ± 13.8	**209.3 ± 14.8**	364.9 ± 18.3
		Our attack	197.2 ± 10.9	196.3 ± 16.1	138.0 ± 14.7	98.5 ± 11.8	107.6 ± 12.3	247.8 ± 15.3
	FCP	No attack	456.9 ± 27.3	**460.4 ± 9.5**	353.7 ± 12.8	340.9 ± 19.0	**351.8 ± 15.5**	**759.0 ± 36.8**
		Random	**389.8 ± 9.2**	**333.8 ± 11.8**	**219.2 ± 12.5**	**171.3 ± 11.5**	178.6 ± 13.6	**467.7 ± 24.2**
		Our attack	**308.6 ± 12.3**	**206.3 ± 10.1**	**178.9 ± 12.3**	**120.7 ± 13.0**	58.3 ± 9.3	**289.4 ± 24.3**



(a) Coordination Ring (b) Cross (c) Double Rings

(d) Double Counters (e) Matrix (f) Clear Division

Figure 3: Training curves of SP agents with BAT and naive adversarial training.



(a) Coordination Ring (b) Cross (c) Double Rings

(d) Double Counters (e) Matrix (f) Clear Division

Figure 4: Training curves of FCP agents with BAT and naive adversarial training.

REFERENCES

Micah Carroll, Rohin Shah, Mark K Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the utility of learning about humans for human-ai coordination. In *Advances in neural information processing systems*, volume 32, 2019.

Ilya Kostrikov. Pytorch implementations of reinforcement learning algorithms. `https://github.com/ikostrikov/pytorch-a2c-ppo-acktr-gail`, 2018.