Appendix

A Additional Results

In this Appendix, we present additional results with SAFER.

Cumulative Safety Violation Graphs In the main paper, we presented the cumulative safety violations at the end of training. Here, we present graphs of the cumulative safety violations in figure 6 throughout training for the baselines and SAFER. In these graphs, we see that SAFER is consistently the safety method throughout training.

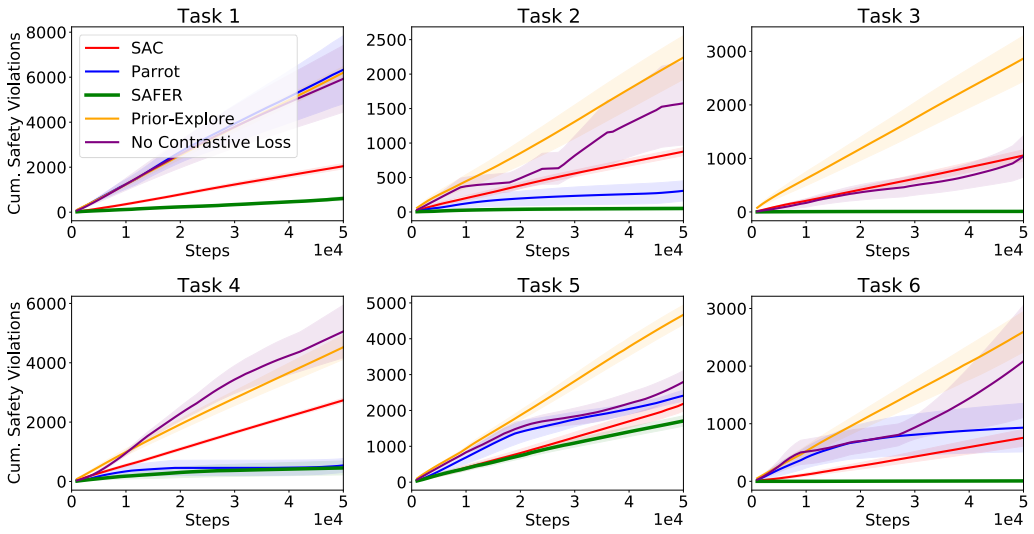

Figure 6: **The cumulative safety violations throughout training** for SAFER and the baselines. We see that SAFER is consistently the safest method throughout training.

Per Step Safety Violations In the main paper, we provide *cumulative* safety violation graphs. Here, we provide the safety violations over the last $1,000$ steps in Figure 7 in order to get a better sense of the safety violations throughout training. We again consistently see SAFER is safety method over the course of training. One interesting observation is that, in Section 2, we discussed how PARROT rates unsafe $(\boldsymbol{s}, \boldsymbol{a})$ pairs as high likelihood. Because PARROT draws on higher likelihood actions from the prior earlier in training, we would expect that PARROT would be more unsafe earlier in training. Empirically, we see this to be the case. Looking at the graphs, PARROT has high safety violation spikes at the beginning of training. These results demonstrate that our earlier observations surrounding the unsafety of PARROT hold true when running RL.

Impact of Probabilistic Treatment One question worth considering is how necessary is it to treat SAFER as a latent variable model and optimize the posterior over the safety variable using variational inference, as is proposed in Section 3.2. It could be easier to treat $\boldsymbol{c}$ as a vector (without defining it as a Guassian random variable), exclude the KL term from Equation 6, and optimize $q_\rho(\boldsymbol{c}|\Lambda)$ with the rest of the objective. To assess whether this is the case, we ran a sweep across different hyperparameter configurations, including the number of bijectors in the real NVP model, the learning rate, λ, and the number of hidden units in each bijector. Doing this, however, we find SAFER quickly diverges, indicating the probabilistic treatment greatly helps stabilize training and is necessary for the success of the method.

Training SAFER Without the Safety Context Variable As an abalation in the main paper, we considered training SAFER without the SAFETY context variable and found that it led to worse

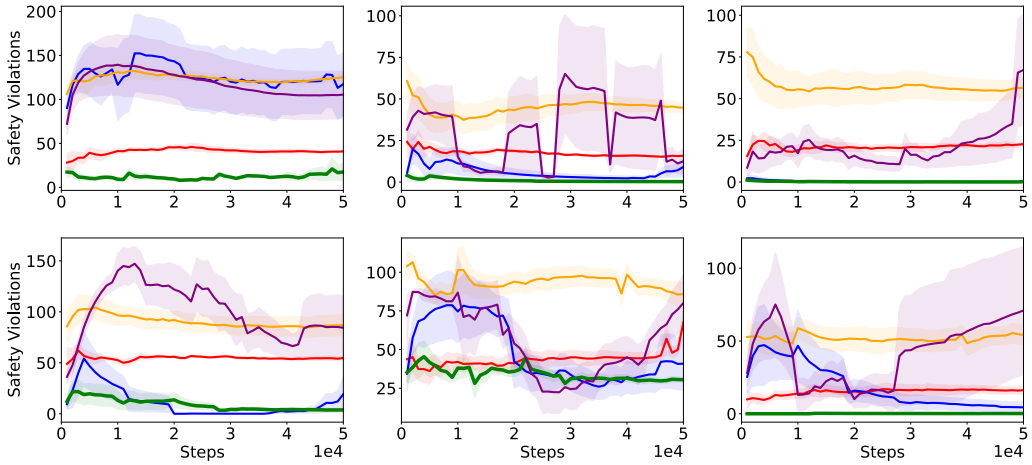

Figure 7: The safety violations over each step of training for each of the tasks (same task ordering as Figure 6). We see that SAFER is consistently the most safe method throughout training.

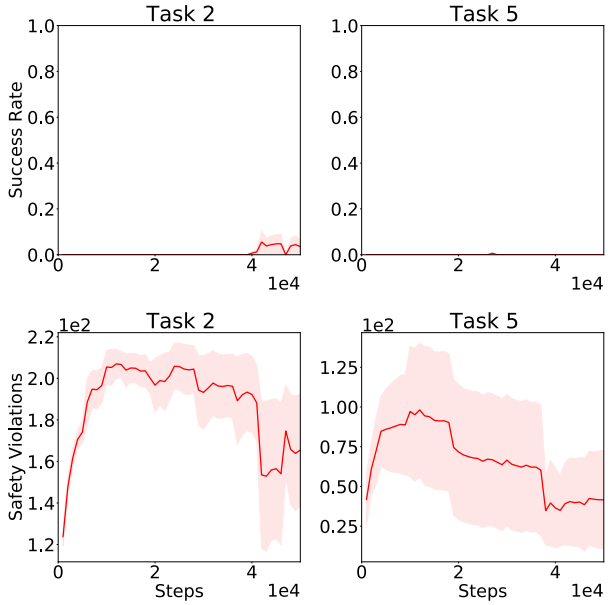

Figure 8: Effectiveness of RL Training using the SAFER objective *without* the safety variable. We see the prior without the safety variable is quite unsuccessful, indicating that the safety variable is critical to enabling the behavioral prior to promote both safe and successful learning.

success rate and relatively higher safety violations. In this Appendix, we provide the full training results in Figure 8 in terms of success rate and per step safety violations. Here, we see that for the tasks considered, training SAFER without the safety variable leads to worse success rates and less safety (compared to the per step success rates in Figure 7.

Training PARROT With Unsafe Data In the main paper, we performed experiments with a PARROT model that was trained with safe data. Meaning, each $w(\boldsymbol{s}, \boldsymbol{a}) = 0$ for each training point. We additionally limited the data to only those tuples that were successful, in order to promote PARROT acquiring safe and successful behaviors. Though it makes the most sense to train PARROT for safety concerned tasks in this fashion, it is worth considering what would happen if we also included unsafe data from successful trajectories. To assess what would happen, we train PARROT using both safe and unsafe data from successful trajectories, using the hyperparameters for PARROT in Sec-

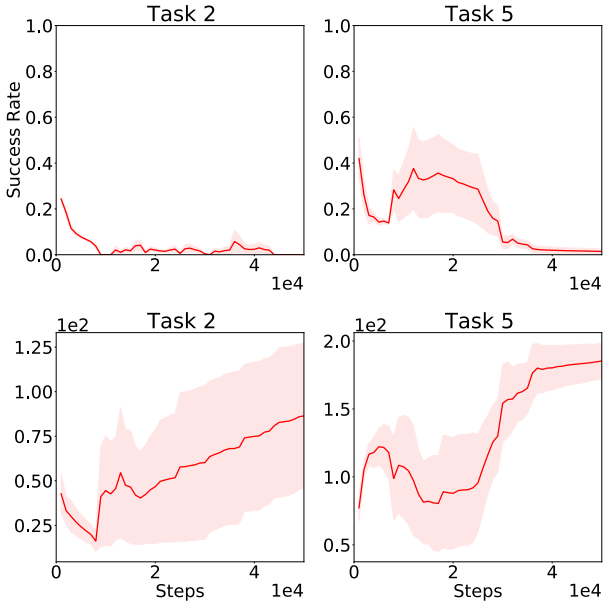

Figure 9: **Training PARROT using unsafe data** from successful trajectories as well as safe data. We see that this leads to leads to relatively worse success rates (top row) as well as relatively higher per step safety violations (bottom row). These results suggest it is best to train PARROT with safe and successful data only.

tion B. The results given in Figure 9 demonstrate that this leads to relatively higher per step safety violations, indicating that it is best to train PARROT with *only* safe data from successful trajectories.

B SAFER HYPERPARAMETER DETAILS

Hyperparameter Details We explored a number of different parameter configurations with SAFER. We tuned λ $(1e-4, 1e-5)$, the number of bijectors in the real NVP flow model $(3, 5)$, the number of components in the context variable $\boldsymbol{c}$ $(8, 32, 64)$, the size of the states window w $(16, 32)$, the optimizer (Adam, SGD+Momentum), and the learning rate $(1e-4, 5e-5)$. We trained for 500k steps and found that using a smaller number of components in the context variable led to more stable training (8). Setting the learning rate to $(1e-4)$ led to much quicker convergence, without sacrificing much stability. Furthermore, training with Adam led to divergence in some cases while SGD+Momentum tended to diverge less often. Between the other parameters considered, there was relatively little difference, and therefore we used a model with learning rate $1e-4$, 3 bijectors, 8 components, 16 states window size, and SGD+Momentum.

C BASELINE METHODS

We select several baseline methods to compare with SAFER. Some of these methods, e.g., PARROT, Prior-Explore, also leverage action primitives trained with offline data to improve efficiency. While we are aware of additional baseline methods, e.g., TrajRL (Shankar and Gupta, 2020; Fox et al., 2017b), HIRL (Ghadirzadeh et al., 2020), in the literature, we omit their comparisons here because it has been shown in prior work (Singh et al., 2021) that their performance is consistently below that of the state of the art.

Soft Actor Critic: Soft-actor critic (SAC) (Haarnoja et al., 2018) is one of the standard model-free policy-gradient based RL methods. Here without using any action primitives we apply SAC to learn a policy that directly maps states in $\mathcal{X}$ to actions in $\mathcal{A}$. Later we also use SAC in all our action primitive based RL methods (e.g., SAFER, PARROT) to optimize the high-level policy while having the low-level policy to be the behavior priors. Therefore, one can view the SAC baseline as one ablation study as well. We use the implementation from TF-Agents (Guadarrama et al., 2018).

We used SAC with autonmatic entropy tuning and tune the number of target network update period, discount factor, policy learning rate, and Q-function learning rate.

PARROT: We compare against the state-of-the-art behavioral prior based RL method PARROT, proposed by Singh et al. (2021). Similar to SAFER, PARROT leverages a conditional normalizing flow and trains the behavior prior using data from successful rollouts. To enforce safety in the PARROT agent, we additionally limit the training data of its behavior prior to *both* safe and successful rollouts, otherwise PARROT may encourage unsafe behaviors. We tune the number of bijectors in the conditional normalizing flow for PARROT $(5, 3)$, the number hidden units in each bijector layer $(128, 256)$, the learning rate $(1e-4, 5e-5, 1e-5)$, the optimizer (Adam or SGD+Momentum), and train for 500k steps. We find using 3 bijectors with learning rate $1e-4$, and the Adam optimizer works best.

Prior-Explore: We also consider the prior-explore method proposed in Singh et al. (2021) as one of our baseline method. Here the prior-explore policy combines the behavioral prior action policy in Equation 3 with an SAC agent to aid exploration of the RL agent. It selects an action from the prior policy with probability δ and from the SAC agent otherwise. Followed from Singh et al. (2021), we set this probability δ to 0.9 and use behavioral prior policy trained for SAFER.

Contextual PARROT (SAFER Without Contrastive Loss): As one ablation study we consider SAFER *without* the contrastive loss. This setup also models the behavioral prior policy with a conditional normalizing flow and the latent safety variable but trains that only with safe and successful data. Note that this baseline method is equivalent to PARROT, with a policy that is a function of the latent safety variable. We use the same parameters as PARROT with this baseline and 8 components in safety variable because we found this number of components to be the most successful with SAFER.

D TRAINING SAFER

In this appendix, we provide psuedo code for the SAFER training procedure in Algorithm 3.

Algorithm 3 SAFER Training

Require: SAFER Behavioral Prior f_ϕ, Safety Variable Posterior $q_\rho(\boldsymbol{c}|\Lambda)$, safe dataset $\mathcal{D}_{\text{safe}}$, unsafe dataset $\mathcal{D}_{\text{unsafe}}$, Steps N, λ

Let `flow_loss(·)` refer to Equation 3

for $n = 1, ..., N$ **do**

$(\boldsymbol{s}, \boldsymbol{a}, \Lambda)_{\text{Safe}} \sim \mathcal{D}_{\text{Safe}}$ ▷ `Sample safe + unsafe batches of data`

$(\boldsymbol{s}, \boldsymbol{a}, \Lambda)_{\text{Unsafe}} \sim \mathcal{D}_{\text{Unsafe}}$

$\boldsymbol{c}_{\text{Safe}} \sim q_\rho(c|\Lambda_{\text{Safe}})$ ▷ `Sample safety variables`

$\boldsymbol{c}_{\text{Unsafe}} \sim q_\rho(c|\Lambda_{\text{Safe}})$

$\mathcal{L}_{\text{safe}} \leftarrow \log\left(\texttt{flow\_loss}(\boldsymbol{s}_{\text{safe}}; \boldsymbol{a}_{\text{safe}}; \boldsymbol{c}_{\text{safe}})\right)$ ▷ `Compute log-likelihoods`

$\mathcal{L}_{\text{unsafe}} \leftarrow \log\left(\texttt{flow\_loss}(\boldsymbol{s}_{\text{unsafe}}; \boldsymbol{a}_{\text{unsafe}}; \boldsymbol{c}_{\text{unsafe}})\right)$

$D_{\text{KL}}^{\text{Safe}} \leftarrow D_{\text{KL}}\left(q_\rho(\boldsymbol{c}|\Lambda_{\text{Safe}})||p(\boldsymbol{c})\right)$ ▷ `Compute KL of safety variables`

$D_{\text{KL}}^{\text{Unsafe}} \leftarrow D_{\text{KL}}\left(q_\rho(\boldsymbol{c}|\Lambda_{\text{Unsafe}})||p(\boldsymbol{c})\right)$

$NLL \leftarrow -(\mathcal{L}_{\text{safe}} - \lambda \cdot \mathcal{L}_{\text{unsafe}} - D_{\text{KL}}^{\text{Safe}} - D_{\text{KL}}^{\text{Unsafe}})$

Minimize NLL and update ϕ, ρ ▷ `Update SAFER`

end for

Return: SAFER Behaviors Prior f_ϕ, Safety Variable Posterior $q_\rho(\boldsymbol{c}|\Lambda)$

E ADDITIONAL TASK EXAMPLES

In this Appendix, we provide additional examples of the tasks included in the safe robotic grasping environment in Figure 10.

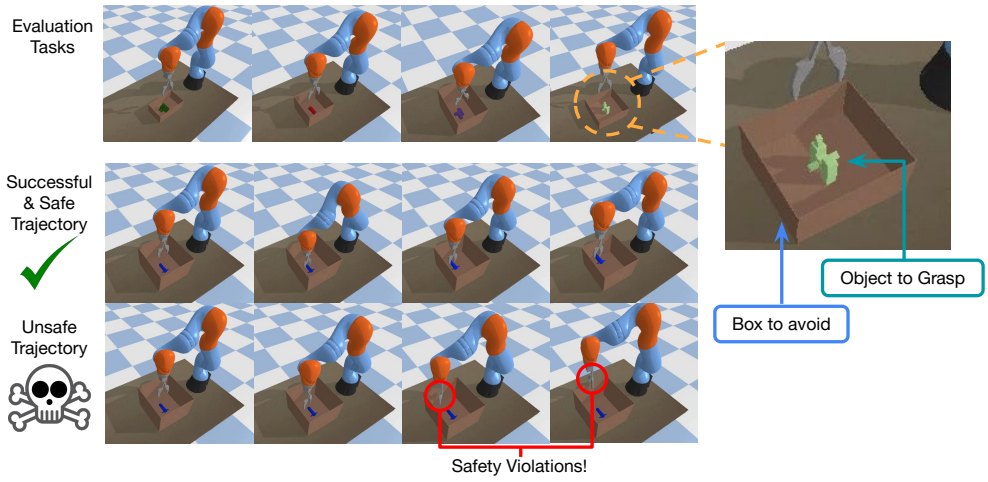

Figure 10: Additional examples of tasks included in the safe robotic grasping environment (top row). The tasks all use different sizes containers, to represent different difficulties in preserving safe behavior. We also provide a zoomed in version of the task (right hand side). Finally, we also include the examples of safe and unsafe trajectories provided in the main paper (Figure 3) for completeness